

WRDC-TR-90-8007
Volume VI
Part 4

AD-A248 908

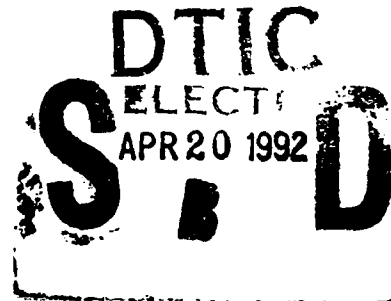


INTEGRATED INFORMATION SUPPORT SYSTEM (IISS)
Volume VI - Network Transaction Manager Subsystem
Part 4 - Network Transaction Manager (NTM) System Programmer's
Manual

S. Barker

Control Data Corporation
Integration Technology Services
2970 Presidential Drive
Fairborn, OH 45324-6209

September 1990



Final Report for Period 1 April 1987 - 31 December 1990

Approved for Public Release; Distribution is Unlimited

92-09952



MANUFACTURING TECHNOLOGY DIRECTORATE
WRIGHT RESEARCH AND DEVELOPMENT CENTER
AIR FORCE SYSTEMS COMMAND
WRIGHT-PATTERSON AIR FORCE BASE, OHIO 45433-6533

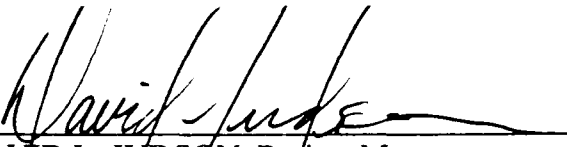
92 4 17 102

NOTICE

When Government drawings, specifications, or other data are used for any purpose other than in connection with a definitely related Government procurement operation, the United States Government thereby incurs no responsibility nor any obligation whatsoever, regardless whether or not the government may have formulated, furnished, or in any way supplied the said drawings, specifications, or other data. It should not, therefore, be construed or implied by any person, persons, or organization that the Government is licensing or conveying any rights or permission to manufacture, use, or market any patented invention that may in any way be related thereto.

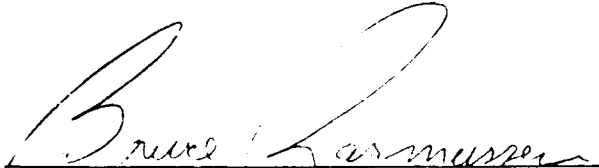
This technical report has been reviewed and is approved for publication.

This report is releasable to the National Technical Information Service (NTIS). At NTIS, it will be available to the general public, including foreign nations


DAVID L. JUDSON, Project Manager
WRDC/MTI
Wright-Patterson AFB, OH 45433-6533

25 July 91
DATE

FOR THE COMMANDER:


BRUCE A. RASMUSSEN, Chief
WRDC/MTI
Wright-Patterson AFB, OH 45433-6533

25 July 91
DATE

If your address has changed, if you wish to be removed from our mailing list, or if the addressee is no longer employed by your organization please notify WRDC/MTI, Wright-Patterson Air Force Base, OH 45433-6533 to help us maintain a current mailing list.

Copies of this report should not be returned unless return is required by security considerations, contractual obligations, or notice on a specific document.

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION Unclassified			1b. RESTRICTIVE MARKINGS None	
2a. SECURITY CLASSIFICATION AUTHORITY			3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for Public Release; Distribution is Unlimited.	
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE				
4. PERFORMING ORGANIZATION REPORT NUMBER(S) SUM 620342000			5. MONITORING ORGANIZATION REPORT NUMBER(S) WRDC-TR-90-8007 Vol. VI, Part 4	
6a. NAME OF PERFORMING ORGANIZATION Control Data Corporation; Integration Technology Services		6b. OFFICE SYMBOL (if applicable) WRDC/MTI	7a. NAME OF MONITORING ORGANIZATION WRDC/MTI	
6c. ADDRESS (City, State, and ZIP Code) 2970 Presidential Drive Fairborn, OH 45324-6209			7b. ADDRESS (City, State, and ZIP Code) WPAFB, OH 45433-6533	
8a. NAME OF FUNDING/SPONSORING ORGANIZATION Wright Research and Development Center, Air Force Systems Command, USAF		8b. OFFICE SYMBOL (if applicable) WRDC/MTI	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUM. F33600-87-C-0464	
8c. ADDRESS (City, State, and ZIP Code) Wright-Patterson AFB, Ohio 45433-6533			10. SOURCE OF FUNDING NOS.	
11. TITLE (Include Security Classification) See Block 19			PROGRAM ELEMENT NO. 78011F	TASK NO. 595600
			WORK UNIT NO. F95600	20950607
12. PERSONAL AUTHOR(S) Structural Dynamics Research Corporation: Barker, S.				
13a. TYPE OF REPORT Final Report	13b. TIME COVERED 4/1/87-12/31/90	14. DATE OF REPORT (Yr., Mo., Day) 1990 September 30	15. PAGE COUNT 75	
16. SUPPLEMENTARY NOTATION WRDC/MTI Project Priority 6203				
17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify block no.)	
FIELD	GROUP	SUB GR.		
1308	0905			
19. ABSTRACT (Continue on reverse if necessary and identify block number)				
<p>This manual offers an understanding of the internal structure of the Network Transaction Manager (NTM) code. It includes descriptions of the architecture of the NTM, its major components, and the modules within those components.</p> <p>Block 11 (Cont) - INTEGRATED INFORMATION SUPPORT SYSTEM (IISS) Vol VI - Network Transaction Manager Subsystem Part 4 - Network Transaction Manager (NTM) System Programmer's Manual</p>				
20. DISTRIBUTION/AVAILABILITY OF ABSTRACT UNCLASSIFIED/UNLIMITED x SAME AS RPT. DTIC USERS			21. ABSTRACT SECURITY CLASSIFICATION Unclassified	
22a. NAME OF RESPONSIBLE INDIVIDUAL David L. Judson			22b. TELEPHONE NO. (Include Area Code) (513) 255-7371	22c. OFFICE SYMBOL WRDC/MTI

FOREWORD

This technical report covers work performed under Air Force Contract F33600-87-C-0464, DAPro Project. This contract is sponsored by the Manufacturing Technology Directorate, Air Force Systems Command, Wright-Patterson Air Force Base, Ohio. It was administered under the technical direction of Mr. Bruce A. Rasmussen, Branch Chief, Integration Technology Division, Manufacturing Technology Directorate, through Mr. David L. Judson, Project Manager. The Prime Contractor was Integration Technology Services, Software Programs Division, of the Control Data Corporation, Dayton, Ohio, under the direction of Mr. W. A. Osborne. The DAPro Project Manager for Control Data Corporation was Mr. Jimmy P. Maxwell.

The DAPro project was created to continue the development, test, and demonstration of the Integrated Information Support System (IISS). The IISS technology work comprises enhancements to IISS software and the establishment and operation of IISS test bed hardware and communications for developers and users.

The following list names the Control Data Corporation subcontractors and their contributing activities:

<u>SUBCONTRACTOR</u>	<u>ROLE</u>
Control Data Corporation	Responsible for the overall Common Data Model design development and implementation, IISS integration and test, and technology transfer of IISS.
D. Appleton Company	Responsible for providing software information services for the Common Data Model and IDEF1X integration methodology.
ONTEK	Responsible for defining and testing a representative integrated system base in Artificial Intelligence techniques to establish fitness for use.
Simpact Corporation	Responsible for Communication development.
Structural Dynamics Research Corporation	Responsible for User Interfaces, Virtual Terminal Interface, and Network Transaction Manager design, development, implementation, and support.
Arizona State University	Responsible for test bed operations and support.

TABLE OF CONTENTS

		Page
SECTION 1	INTRODUCTION	1-1
SECTION 2	SYSTEM OVERVIEW.	2-1
2.1	The NTM Within the IISS Architecture . .	2-1
2.2	NTM Components - Overview.	2-3
2.2.1	Monitor Application Process.	2-3
2.2.2	Message Processing Unit.	2-4
SECTION 3	LOGICAL PARTITIONING OF SOFTWARE	3-1
3.1	Monitor Application Process.	3-1
3.1.1	IISS Start-up Processing	3-2
3.1.2	System Operations Processing	3-3
3.1.2.1	Message Event Processing	3-3
3.1.2.2	Console Event Processing	3-4
3.1.3	IISS Shut-down Processing.	3-4
3.1.4	Monitor Diagnostics.	3-5
3.1.5	Configuring Monitor AP	3-5
3.2	Message Processing Unit.	3-5
3.2.1	Logical Structure of the MPU Code. . .	3-5
3.2.2	MPU Start-up Processing.	3-6
3.2.3	MPU Operations Processing.	3-8
3.2.3.1	Message Event Processing	3-8
3.2.3.1.1	Manage Message	3-8
3.2.3.1.2	Process Message.	3-9
3.2.3.2	Timer Event Processing	3-11
3.2.3.3	Guaranteed Delivery Processing . . .	3-12
3.2.4	Shut-Down Phase of the MPU	3-13
3.2.5	Error Handling by the Message Processing Unit.	3-14
3.2.6	Debugging the Message Processing Unit.	3-14
3.3	NTM Services	3-14
3.3.1	NTM Initiation Services.	3-28
3.3.1.1	INITAL	3-20
3.3.1.1.1	Creating the AP's Input Mailboxes.	3-20
3.3.1.1.2	Initializing AP Global Data Structures	3-20
3.3.1.1.3	Informing the NTM That the AP Is Alive.	3-21
3.3.1.1.4	Receiving System-State Informati from the NTM	3-21
3.3.1.2	INITEX	3-21
3.3.1.3	INICOM	3-21
3.3.1.4	LOGON.	3-22
3.3.2	NTM Communication Services	3-22
3.3.2.1	NSEND.	3-23
3.3.2.2	ISEND.	3-24
3.3.2.3	QSEND.	3-24
3.3.2.4	RVC.	3-25
3.3.2.5	CHKMSG	3-27
3.3.2.6	ISTMOD	3-28
3.3.2.7	SIGERR	3-29

TABLE OF CONTENTS (Continued)

		<u>Page</u>
3.3.3	NTM Status Messages.	3-28
3.3.3.1	CHGROL	3-28
3.3.3.2	GETUSR	3-29
3.3.3.3	WHTHST	3-29
3.3.4	NTM Termination Services	3-30
3.3.4.1	LOGOFF	3-30
3.3.4.2	TRMNAT	3-30
3.4	NTM Internal Table Access and Control . .	3-31
SECTION 4	DATA STRUCTURES.	4-1
4.1	Sysgen Data Processing	4-1
4.2	Monitor Application Process.	4-7
4.2.1	Monitor AP Internal Table.	4-7
4.3	Message Processing Unit.	4-7
4.3.1	Queues	4-7
4.3.1.1	Off-Cluster Message Queue.	4-7
4.3.1.2	On-Cluster Message Queue	4-7
4.3.2	APC Initialization Data.	4-7
4.4	NTM Internal Tables.	4-9
SECTION 5	HOST SYSTEM DEPENDENT SPECIFICS.	5-1
5.1	Introduction	5-1
5.2	VAX Implementation of System Support Primitives	5-2
5.2.1	Process Control Modules.	5-2
5.2.1.1	CRTPRC	5-2
5.2.1.2	DELPRC	5-3
5.2.1.3	GETNAM	5-4
5.2.1.4	SGTPNM	5-5
5.2.2	Global Memory Routine.	5-5
5.2.2.1	CGSHST	5-9
5.2.2.2	MAPHSI	5-9
5.2.3	System Time Routines	5-10
5.2.3.1	ASCTIM	5-10
5.2.3.2	GETTIM	5-10

APPENDICES

APPENDIX A	TERMS AND ABBREVIATIONS.	A-1
APPENDIX B	NTM DOCUMENTATION.	B-1

LIST OF ILLUSTRATIONS

<u>Figure</u>	<u>Title</u>	<u>Page</u>
2-1	IISS Architecture - Conceptual Model on VAX. . .	2-2
3-1	Monitor AP Five Major Modules.	3-1
3-2	MPU Logical Structure.	3-6
3-3	AP Mailboxes	3-16
3-4	NTM Services	3-17
3-5	IISS AP Code Unit.	3-17
3-6	INITAL, INITEX, and INICOM	3-19
3-7	RCV Condition Matrix	3-26
4-1	Monitor's Default Sysgen Data.	4-2
4-2	MPU's Default Sysgen Data.	4-3
4-3	SYSGEN Data Items Modified or Added via MPUGEN .	4-6
4-4	NTM Tables - Overview.	4-10
4-5	NTM Tables by Data "Stability"	4-12

SECTION 1

INTRODUCTION

This NTM System Programmer's Manual is offered to provide an understanding of the internal structure of the Network Transaction Manager (NTM) code. To place the code in its proper context, Section 1 provides an overview of the Architecture of the NTM and the major components within that architecture. Section 2 discusses the details of the modules within each of the major components. Section 3 discusses the Data Structures associated with the major components along with a discussion of the NTM internal tables. Section 4 covers those aspects of the NTM code that are host system dependent.

Accession For	
NTIS GRA&I	<input checked="checked" type="checkbox"/>
DTIC TAB	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
By	
Distribution	
Availability Codes	
Dist	Avail and/or Special
A-1	

SECTION 2

SYSTEM OVERVIEW

2.1 The NTM Within the IISS Architecture

The IISS is a system that incorporates heterogeneous host machines into a network to provide transaction processing services within a manufacturing environment. Figure 2-1 provides the conceptual view of the IISS on one host machine; in this instance, the VAX.*

The basis of the IISS architecture is the concept of AP clusters. An AP cluster is defined as a highly cohesive group (one or more) of Application Processes that have either one or no Database Manager (and by extension, database) in common. This concept allows for the isolation of application functions based upon the data they need to access. The IISS components, the User Interface (UI), Communications Handler (COMM), and Common Data Model Request Processor (CDMRP) are seen as Application Processes that each reside on a distinct AP cluster. Non-integrated or "Existing" Applications (applications designed without adherence to IISS integration rules) may also reside on separate AP clusters in accordance with the rule that they access the cluster's common database. Integrated or "New" Applications (applications designed and written in accordance with the IISS Integration Rules) will either reside on separate AP clusters or be added to existing AP clusters depending upon database(s) they need to access. An Application Process (AP) within the IISS architecture is a cohesive unit of software that can be initiated as a unit to perform some function or functions. All APs within the IISS are treated in the same manner by the NTM.

The NTM provides the common operating thread for each AP cluster. A component of the NTM known as the Message Processing Unit (MPU) will be associated with each AP cluster to provide for message management, process management, and to maintain AP cluster operability.

* VAX is a trademark of the Digital Equipment Corporation.

Some examples of APs in the IISS are: User Application Processes such as an MCOMM component capable of processing a transaction; and IISS components such as the User Interface, Communication Handler, and Common Data Model Request Processor. The NTM's concern in all cases is to provide a transparent logical link from any AP to any other AP in the IISS system. Each AP is seen by the NTM as a stand-alone unit that is fully capable of performing its own processing and normally terminating itself.

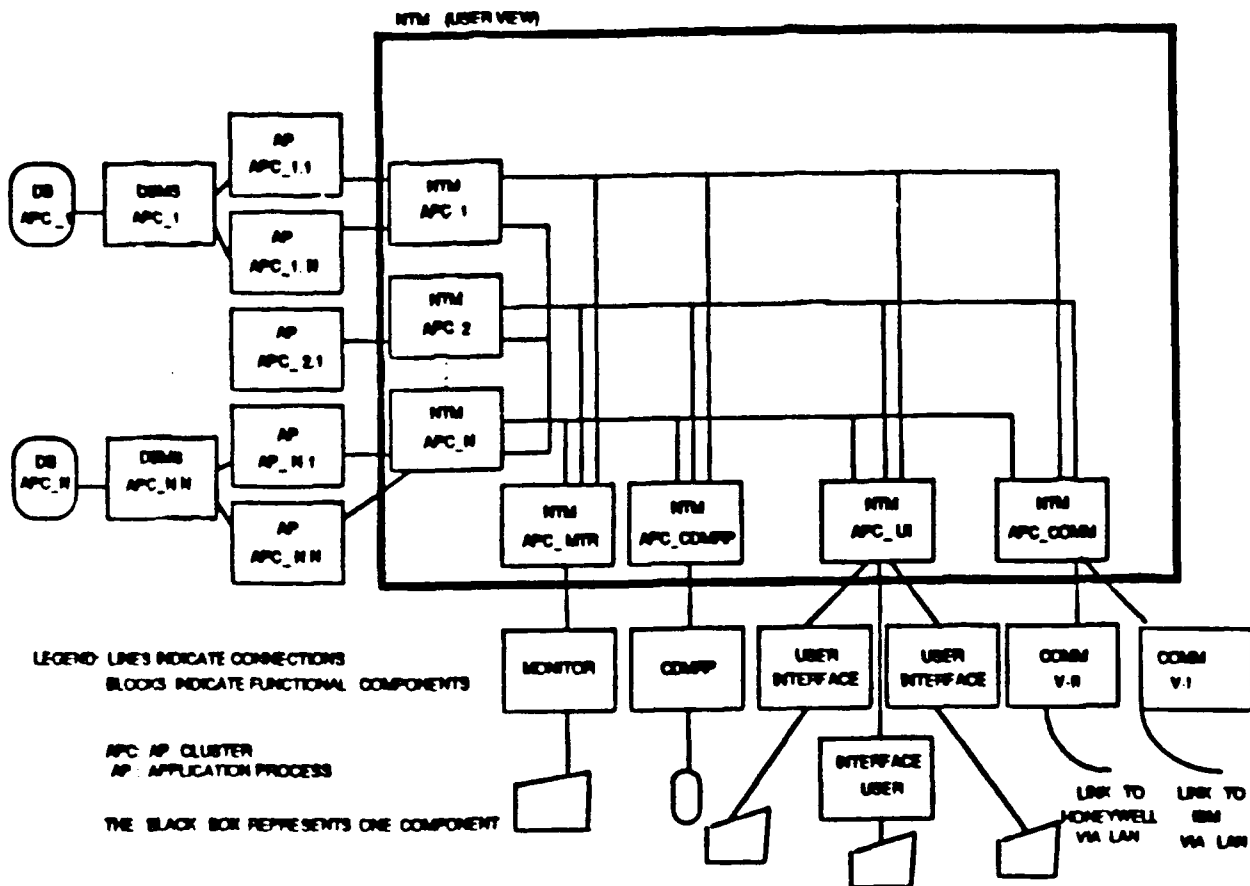


Figure 2-1. IISS Architecture - Conceptual Model on VAX

In addition to the MPU component, the NTM contains the Monitor Application Process and the NTM system and table services. These three components work together to provide system start-up and shut-down coordination, system monitoring, IISS Operator Interface, and message processing services for the APs.

2.2 NTM Components - Overview

2.2.1 Monitor Application Process

The Monitor Application Process (Monitor AP) is the NTM component which provides a central control point for the IISS environment on an IISS host node. Monitor AP's responsibilities include coordinating IISS start-up and shut-down, monitoring IISS operating activity, and providing an interface to the IISS operator.

The Monitor AP is the first NTM component to begin execution when the IISS is started. All Application Process Clusters (APC's) and all IISS network communications are initiated under the control of the Monitor AP. Monitor AP begins execution by initializing its local variables and creating and initializing the host global NTM tables. It then spawns its own MPU and establishes communications with it via a message exchange protocol. With the complete Monitor APC active, the Monitor AP begins requesting the initiation of host local APCs. Messages are sent to the Monitor APC MPU which interprets these commands and spawns the requested MPU. Once all local APCs have been started, network communications are initiated. Monitor AP accomplishes this by first requesting that the COMM APC MPU initiate the appropriate COMM AP and then forwarding a "Start Link" message to the COMM AP.

When start-up is complete, Monitor AP updates the system tables and sends a "Host Active" message to all local MPU's indicating that normal IISS processing may begin.

The process described above is currently applicable only to the "central" node. A "remote" host has a slightly different start-up procedure. After initiating the Monitor APC, as above, the remote node will attempt to establish a communication link to the central node before continuing with start-up. The remote host's Monitor AP initiates its COMM APC and the appropriate COMM link and waits for a response from the central node Monitor AP indicating that it is active. Once this link has been established, start-up continues as described above.

With the IISS started and in a stable state, Monitor AP assumes the role of system monitor-handling system error messages, processing operator commands, and maintaining the operability of the IISS environment. The Monitor AP accepts error and status messages from system components for display on the operator console. The following operator commands are supported by the Monitor AP:

1. Shut-down IISS
2. Shut-down APC
3. Start APC
4. Shut-down COMM Link
5. Start COMM Link
6. Cancel IISS shut-down
7. Display System Status
8. Display Active AP's
9. Enable SIGERR Messages
10. Disable SIGERR Messages
11. Select Logging Features
12. Start New Log File

Upon receiving the Shut-down IISS command, the Monitor begins coordinating the orderly shut-down of the IISS. Prior to actually beginning shut-down processing, the Monitor AP notifies IISS users that the system shut-down is pending via the UI APC's MPU. When shut-down begins, the non-component APCs are shut down first. Network communications are then terminated followed by the shut-down of the component APCs. When all other NTM components are shut down, the Monitor AP requests the shut-down of its MPU and then ends its own execution.

2.2.2 Message Processing Unit

The purpose of the Message Processing Unit (MPU) of the Network Transaction Manager (NTM) is to handle communications between application processes on the IISS. These processes are associated with a cluster with each cluster having its own Message Processing Unit. The MPU enables its application processes to communicate with and initiate other application processes on the cluster, off the cluster, and off host via messages. The MPU monitors its AP's activities and provides other useful features (such as Message Pairing, tracking an AP's "children," and queuing messages) for the APs.

The Message Processing Unit has three stages of operation: Start-up, APC Operation, and Shut-down. The Monitor Application Process (Monitor AP) initiates the first instance of an MPU (the Monitor MPU) while all other Message Processing Units are initiated by the Monitor MPU.

The Start-up stage of an MPU involves getting the MPU's process name from the operating system, reading the start-up file for sysgen data and creating the MPU's high and low priority mailboxes. The MPU then sends a message to the Monitor AP requesting the status of the tables on the cluster and waits for a response from the Monitor AP. If the response does not arrive within a given time, the MPU will send an "AP cluster terminating" message to the Monitor AP and will proceed to terminate. Should the message arrive, the MPU will populate the tables in accordance with the current table status. The MPU then sends an "AP cluster alive" message to the Monitor AP and waits for final start-up instructions.

Should an error occur during the IISS start-up, the Monitor AP will instruct each Message Processing Unit to terminate. If IISS start-up is successful, the Monitor AP will send a "host active" message to signal the MPU that Start-up is complete. If the final instruction message does not arrive within a given time, the MPU will assume an error has occurred and begin the AP cluster termination process on its own.

During the MPU Operations phase, the MPU is available to application processes on its MPU. The APs can communicate with one another and initiate other application processes by sending messages through the MPU. The MPU processes these messages in two phases; Manage Message and Manage Process.

The Manage Message phase authorizes the message, verifies the message category, assigns a serial number and completes the message header for messages from on the cluster. If the message is from another cluster, Manage Message verifies that the message arrived at the correct destination. If a message error occurs, the MPU will send an error message to the Source AP. If a system error occurs, the MPU will send an error message to the Monitor AP. All messages are currently logged in the Manage Message phase. The type of processing to be done on the message is also determined at this time. Messages destined to off-host AP's are sent to the Communication Application's MPU; messages destined to on-host but off-cluster are delivered to the off-cluster MPU; and messages destined for on-cluster are processed by phase two of MPU operations; Manage Process.

Manage Process handles each message destined for the AP cluster according to its message category and type. Some messages cause the initiation of other application processes in the IISS; others are simply delivered to the Destination AP.

During the MPU Operations phase, a timer is set for each MPU. When the timer goes off, a checking routine is invoked. The checking routine looks at time dependent processes such as the AP initiation, message pairing, and message queuing services. Tables and queues are checked for messages that have timed out, AP's that have completed their initiation phase, messages waiting for a resource, and so forth. Each entity is handled in accordance with its current status. As an example, the Message Pair table has an entry for each message that requires a response. If the time-out time on the entry has passed (compared to the system time), the MPU deletes the entry and sends a "time-out error" message to the Source AP. After completion of the checking routine, the timer is reset.

In addition to handling application process messages, each MPU has a high-priority mailbox from which it reads system command messages from the IISS operator via the Monitor AP. For example, the operator may request a list of all active APs on the MPU's cluster. The system command message that requires the MPU to go into its final stage of operation is a "Shut-down AP Cluster" message from the operator via the Monitor AP.

The Shut-down stage of the Message Processing Unit operate operates in a similar manner to the MPU Operations stage. Messages are still processed by the MPU. However, when the MPU receives the "Shut-down APC" message, it looks at each application process in the AP Status table and, according to the AP's characteristic, aborts the AP or informs it that it must shut itself down. The MPU then checks its mailboxes for a message from each of the APs that were told to shut-down. Only AP Status messages and system commands from the Monitor AP are processed normally during the shut-down phase. All other types of messages may be logged but otherwise ignored by the MPU. The MPU systematically checks to see if all the application processes on the cluster are terminated. If no more messages are in the mailboxes to be processed and all the APs are no longer running, the MPU invokes its final shut-down processing. The AP cluster initialization data is saved in a local file where it will be used during the next IISS or cluster start-up. The MPU's input mailboxes are deleted and an "APC Terminating" message is sent to the Monitor AP. The Message Processing Unit then ends execution.

SECTION 3

LOGICAL PARTITIONING OF SOFTWARE

3.1 Monitor Application Process

The Monitor AP consists of five major modules which are executed at specific times during the IISS execution. MONITR, the main program of the Monitor AP, calls the modules associated with the five major functions. Each in turn calls a hierarchy of modules which provide the specific functionality of the Monitor AP. The topmost levels of this hierarchy are:

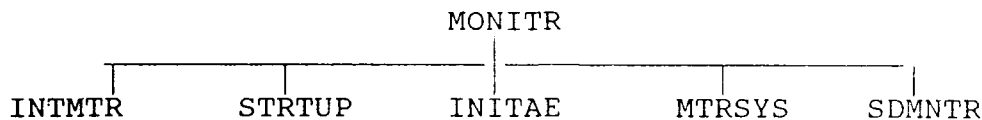


Figure 3-1. Monitor AP Five Major Modules

The functional description of the Monitor AP is available in the NTM Development Specification [1]*. Structure charts and module descriptions are available in the NTM Product Specification, Vol. 1 [2].

The Monitor AP's functions during IISS execution are divided between the IISS Start-up, System Operation, and IISS shut-down phases. INTMTR, and STRTUP control IISS Start-up. INITAE initializes asynchronous processing for MTRSYS. MTRSYS serves as the focal point during System Operation and IISS shut-down. Moni-TR will call SDMNTR to handle the final shut-down processing of the Monitor APC.

[1] Square brackets refer to references in appendix B

* References are made to documents described in Section 5.2.

3.1.1 IISS Start-up Processing

IISS start-up processing begins when the IISS Operator invokes MONITR by whatever means are applicable to the local operating system. MONITR begins processing by reading a sysgen file ** for data required to initialize the Monitor's AP's common data (MTRCMN), establishing the connection to the operator's console, and creating the space for the Global Host Memory. MONITR then calls INTMTR which will coordinate the start-up of the Monitor's APC. INTMTR accomplishes this by calling the next level routines described below.

- o CRTMBX: Creates the Monitor AP's input mailbox.
- o BLDTBL: Initializes and populates the host's NTM global tables.
- o CRTPRC: Spawns the error and message logging tasks.
- o CRTPRC: Spawns the Monitor APC's MPU.
- o PRCTSR: Waits for the table status request message from the monitor APC's MPU. If any other message arrives at this time, it will be processed in accordance with its type. Control will always return to PRCTSR until either the expected message arrives or the TIMEOUT period expires. Upon receipt of the expected message, the table status return is sent to the Monitor APC's MPU.
- o PRCSSR: Waits for the APC alive status message. As in PRCTSR other messages will be processed if they arrive.
- o APCSTS: Called if the APC Alive message arrives before the timeout period expires. Updates the APC status table and displays the APCs one at a time. Calls in for each APC to be started.

**The sysgen file read by the Monitor and the MPU is described in Section 3.1 of this document.

In the case of Monitor's MPU's start-up, an AP Alive message is sent to the MPU in accordance with the MPU-AP Protocol. This allows the MPU to enter the Monitor AP in the AP Status Table to facilitate further message processing.

Upon successful completion of INTMTR, the Monitor AP's MPU is fully active. The next steps are to start the remaining MPUs and establish the network communications link. These steps are coordinated by STRTUP. If the Monitor AP is on a remote node, the link to the central node is established via a call to LKTOCN. In either case, the following routines are called.

- o STLOAC: Coordinates the start-up of the local (on-host) APCs one at a time. Calls NITAC for each APC to be started.
- o INITAC: Sends a Start APC message to the Monitor APs and calls PRCTSR and PRCSSR to handle the message protocol. APCSTS is called on successful start-up. If APC start-up fails, APCSTF is called.
- o STNTCM: Coordinates the start-up of the links to remote hosts. Starts the Communications Handler AP and the Link to remote host. If either message encounters an error, STLNKF is called to display a message for the Operator. The Link status message will arrive and be handled during the system operations phase.
- o STCPLT: Coordinates the remaining start-up processing by sending status messages to remote hosts (if any links have been established) and to the Local APCs.

3.1.2 System Operations Processing

Upon the successful completion of start-up, MONITR calls INITAE to set up the Systems Operations Phase. INITAE issues reads on both the Operator's console and the Monitor AP's input mailbox. The IISS start-up banner is displayed on the console.

MONITR then calls MTRSYS. This routine waits for an event to occur, either at the console or the mailbox. Each event is handled as it occurs with control returning to MTRSYS when the processing is complete. The two major paths of event processing are discussed below.

3.1.2.1 Message Event Processing

All messages arriving before shut-down processing begins trigger a call to PRCMSG which acts as a filter by reading the message type from the header and calling the appropriate message handling routine. Each message type received by the Monitor AP has its own routine. A message having a type not recognized by PRCMSG is displayed on the operator's console.

3.1.2.2 Console Event Processing

All console events trigger a call to PRCCON. As long as shut-down processing has not begun, PRCCON will call XOPCMD to interpret the command. XOPCMD is similar to PRCMSG in that it will read the command and call the appropriate routine. Each command has its own controlling routine.

3.1.3 IISS Shut-down Processing

IISS Shut-down is invoked by an operator command which triggers a console event to MTRSYS. The command interpreter, XOPCMD, calls XSDCMD to coordinate further operator input regarding the amount of time before the actual shut-down begins. XSDCMD obtains the time until shut-down and calls SHTDWN. SHTDWN is the main procedure for the shut-down processing. This processing is partitioned into seven phases as described below

- Phase 1: If time until shut-down is one minute or more, this phase is executed to inform all users on the system that shut-down is pending (TELUSR). The shut-down command may be cancelled only during this phase.
- Phase 2: This phase marks the beginning of actual shut-down processing. During this phase, the local non-component APCs and remote hosts are told to shutdown (SDRHNC). At the end of this phase, control is briefly passed to MTRSYS to wait for status messages.
- Phase 3: When the status messages arrive, MTRSYS calls SHTDWN directly. The messages are processed via a call to SDSTAT. SDSTAT reads the message type and calls the appropriate message handling routine.
- Phase 4: At this point, the network communications links are shut-down via a call to TRNTCM.
- Phase 5: MTRSYS waits for the Link Status messages. These messages are processed by SDSTAT.
- Phase 6: All remaining Local APCs are told to shutdown (SDAL-CP).
- Phase 7: Processes the status returns from Phase 6 in the manner described above for Phases 3 and 5.

At the end of Phase 7 control returns to MONIIR which calls SDMNTR to coordinate the shut-down of Monitor AP's APC. This involves an exchange of shut-down messages between the AP and the MPU. When the MPU terminates, the shut-down complete message is displayed at the console. The input mailbox is deleted and the connection to the operator's console is cancelled.

3.1.4 Monitor Diagnostics

When investigating problems with the Monitor AP, the system programmer should start with one of the high level modules and work downward in the hierarchy. The state that the system is in when the problem occurs can indicate where to begin looking in the code. For instance, if the error occurs prior to notification that the Monitor APC has successfully started, INTMTR is the module to investigate. From the point at which Monitor APC starts until start-up is complete, Monitor AP is under the control of the STRTUP module. If start-up is successful but monitor fails before the NTM banner is displayed to the operator console, INITAE is the place to look. From the time at which the operator is prompted for input until IISS shut-down progresses to the point at which all NTM components have terminated except the Monitor APC, MTRSYS is the module which is executing.

Any errors during INTMTR processing will cause the Monitor AP to terminate. The error code and message displayed to the operator console will indicate the nature of the error (see the IISS Network Transaction Manager Operator's Manual [3] for a description of the NTM error codes).

Certain system errors during STRTUP will cause the Monitor AP to fail. If a component APC fails to initialize properly, the Monitor AP on the node will perform shutdown processing.

3.1.5 Configuring Monitor AP

When Monitor AP is to be configured to run in a new (or changed) IISS environment, it is necessary to update the system configuration variables. These variables are added to (or modified in) the sysgen data file read by Monitor at start-up. The manipulation of this file is discussed in Section 3.1.

Monitor AP has a special internal table which is used to maintain information about the IISS system configuration. A discussion of this table is found in Section 3.2.1 of this document.

3.2 Message Processing Unit

3.2.1 Logical Structure of the MPU Code

The functions of the Message Processing Unit (MPU) component of the NTM can be divided logically between the Start-up, Operation, and Shut-down phases. The Operation phase includes those functions required to handle messages received by the MPU. These messages may have as their source either an on-cluster Application Process or another MPU. The values given in the message header dictate the path to be taken through the code. EXCMPU, the main program of the Message Processing Unit, controls the flow of these MPU phases. Each phase is, in turn, controlled by its own main program. The top of the hierarchy is given in Figure 3-2.

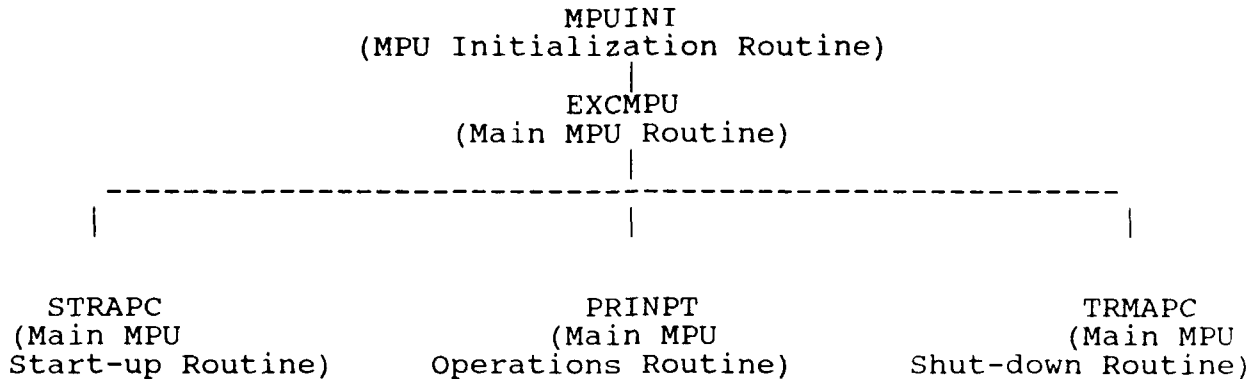


Figure 3-2. MPU Logical Structure

Each of these routines will call lower level routines. While the calling sequences of the start-up and shut-down phases are pre-determined, the calls made during MPU Operations are mostly determined by the nature of the message being processed.

3.2.2 MPU Start-up Processing

An MPU is spawned by the Monitor AP's MPU upon request from the Monitor AP. (The Monitor AP's MPU is spawned directly by the Monitor AP. This is the only deviation in the startup phase of any MPU). Once spawned, MPUINI begins operation by mapping the global host storage to tables within the MPU. The MPUINI routine is system specific due to the method used by the different operating systems in accessing common storage between processes. Mapping to the host tables on a VAX computing system is performed via a call to "MAPHST" (Sec. 4.2.2.3). This routine establishes the MPU's connection to the system global tables. MPUINI will then call EXCMPU to continue startup processing. EXCMPU begins execution by calling "GETNAM" (Section 4.2.1.3) to determine its identity. Using the AP Name returned to it, EXCMPU reads its sysgen initialization file (See Section 3.2 for details). Information from this file is read into local program storage. This area contains:

- o IISS Instance
- o The last serial number assigned to a message (saved from the previous IISS session)
- o The name of the host on which the MPU resides
- o The name and APC location of the host's Monitor AP
- o The name and APC location of the central node's Monitor AP

- o Queue Data (maximum allowed in queue, maximum last key)
- o The name of the CDM's AP Cluster
- o Timer values
- o The name of the Communication Handler's AP cluster
- o The name of the specific MPU's AP cluster
- o The Process Name of the specific MPU

The structure of this file is given in Section 3-2.

Upon successful initialization, EXCMPU is the main MPU processing routine. STRAPC is called by EXCMPU to complete the start-up processing. STRAPC in turn calls four next-level routines to handle remaining start-up processing and message protocol.

- o INITST: Opens the AP cluster's files and logs and creates the APC's hot and cold mailboxes. Upon successful completion, STRAPC sends a Table Status Request (TS) message to the Monitor AP.
- o TABPRC: Populates the APC static Tables* in accordance with the Table Status Return (ST) Message from the Monitor AP. If the status return indicates that the tables have changed, the tables are populated from the CDM. If there is no change, the tables are populated via a series of initialization routines (described in Section 2.4). Local dynamic table headers (for those tables having no population requirement) are also initialized. Upon successful completion, STRAPC sends an APC Alive (LV) message to the Monitor AP.
- o FSTART: Waits for and processes final start-up messages from the Monitor AP. The final start-up message is the Host Active Message. In the case of the Monitor AP's MPU, the Build Table message may also arrive at this time. Upon successful completion, control is returned to EXCMPU and the MPU Operation phase begins.

*These tables are: The Authority Table, the Authority Check Table, the AP Characteristics Table, and the AP Information Table.

- o GDMSGGS: Processes the Guaranteed Delivery Message file to determine if there are any Guaranteed Delivery (GD) messages that need to be processed. The recovery action taken by GDMSGGS is determined by the state status of the GD message in the file.

3.2.3 MPU Operations Processing

The MPU Operations Phase is controlled by PRINPT. As long as the MPU is active, control is passed to this routine upon completion of prior processing.

PRINPT waits on one of three events: a message in the hot mailbox, a message in the cold mailbox, or a timer. Processing is based on the event that occurs first. The routines associated with these events are described below.

3.2.3.1 Message Event Processing

3.2.3.1.1 Manage Message

The arrival of a message in either the hot or cold mailbox starts the MPU's message processing. As long as the MPU is not in shut-down mode (or the message is an AP status message or a system command), MNGMSG is called. This routine examines the message header to determine the next level of processing required and makes the following calls based on the specified conditions.

- o GDMSGGS: Called if the message from off-cluster requires Guaranteed Delivery processing (category "A" messages from off-cluster).
- o PRCONC: Called if the message has as its source an on-cluster AP. PRCONC, in turn, may call the following under the specified conditions:
- o VMSGCT: verifies the message category field if the message header is new.
- o POSAUT: checks the level of authority required by the destination AP. If the destination AP is restricted, a call is made to AUTMSG to verify that the message is authorized for its destination.
- o CMPHDR: if there are no errors on the messages, validates certain header fields and calls MPUINF to supply remaining values.

- o GDMSGs: if the message from on-cluster (category "A") requires guaranteed delivery, creates an entry in the guaranteed delivery table.
- o ADDPR: if the message requires pairing support, creates an entry in the Message Pair table.
- o CHDPRC: if the message can result in the initiation of an AP, creates an entry in the Child Table.
- o SNDSAP: if the source AP is waiting for an acknowledgment and no errors have been encountered, formats and sends the Message ACK. If an error occurs at any point in this message processing, SNDSAP will be used to send the error message to the source AP.
- o VYOFFC: Called if the message has as its source an off-cluster AP and the calling MPU is not on the COMM APC. Verifies that the message has arrived at the correct MPU.
- o RMVAST: Called if the message is from off-cluster and the calling MPU is on the COMM APC. Removes asterisks placed in the header to facilitate COMM processing.

Upon completion of MNGMSG functions, control is returned to PRINPT.

3.2.3.1.2 Process Message

When control is returned to PRINPT and if there are no errors on the message, a call is made to RTESND. This routine examines the message header to determine the next level of processing required. The following calls may be made under the specified conditions.

- o MNCPRC: Called if the message destination is an on-cluster AP. MNGPRC may, in turn, call the following under the specified conditions.
- o APSTAT: handles AP status messages. Based on the status type, will call:
- o IMALIV: upon receipt of AP Alive Message, handles final AP initiation processing.
- o NEXCNA: upon receipt of an AP external connection request message, handles assignment of externally connected application processes.

- o APDEAD: upon receipt of AP Dying Message, handles AP termination processing not covered by the system service TRMNAT.
- o CHDSTM: upon receipt of a Child AP Status message, handles child table updates.
- o DELCLD: upon receipt of an unsuccessful initiation return, deletes the relevant entry from the child table.
- o CHDPRC: upon receipt of an unsolicited initiation acknowledgment, updates an entry in the Child Table.
- o PFINIT: Called if the message is a specific initiation message or where it appears that the message requires the initiation of an AP. This routine verifies that: the initiation is required; the resources are available; and that the AP is not in an abort state. If no errors are encountered, the initiation is performed via a call to either INITAP (where there are no instances of the AP currently running) or PRINIT (where there are instances running, PRINIT may in turn call INITAP if the AP is not running the maximum instances allowed).
- o SYSCOM: Handles NTM system commands (generally messages from Monitor). Based on the received command, this routine may call:
 - o SHUTAP: to shut-down an AP (operator's command).
 - o SHTAPC: to shut-down an APC (operator's command or part of system shut-down).
 - o INITAP: to start an MPU -- this call is made at this point by the Monitor AP's MPU only.
 - o CLNHSD: handles processing required when a remote host shuts down.
 - o LISTPR: processes the request for a list of active APs on-cluster (operator command).

- o GDMMSG: handles processing of Guaranteed Delivery NTM protocol messages (MPU-GDACK, MPU-GDACKBACK, and the ACK from a user AP).
- o CLNUP: processes cleanup command (message from Parent AP's MPU).
- o NABORT: processes Abort AP commands (operator command or request from an AP).
- o SDPEND: process shut-down pending message.
- o CNCLSD: process cancel shut-down message.
- o HSTNRQ: processes requests from the WHTHST service. (What host is this given AP on?)
- o DLVMSG: Called where the message is to be delivered to the AP. This routine calls SENDAP to check the AP's queues for outstanding messages and proceeds to write the message(s) into the correct mailbox until all messages are delivered or the AP's mailbox is full.
- o OFFCLQ: Called if the message destination is off-cluster. If the destination is on-host, this routine checks the status of the destination APC and, if available, writes the message to that MPU's mailbox. If the mailbox is full, the message is queued for later delivery. If the destination is off-host, RTESND adds asterisks to the header and OFFCLQ writes the message to COMM's MPU.

Upon completion of RTSEND functions, control is returned to PRINPT.

3.2.3.2 Timer Event Processing

As previously mentioned, PRINPT waits on a timer event as well as the mailbox events. Upon the occurrence of a timer event, PRINPT will call the TIMCHK routine to perform actions required by the timer. TIMCHK calls the following routines to perform the time-related checks on tables and queues.

- o PAIRCK: Reads the message pair table for messages that have not received a required response in the allotted time. Timed out messages are deleted from the table and error messages are sent.

- o IATCHK: Reads the I'm Alive table for APs that have not sent I'm Alive messages within the allotted time. Where the AP has not responded, its table entry is deleted and an unsuccessful initiation message is sent.
- o SENDAP: Checks for messages queued for delivery to on-cluster APs. If the AP is now available, the message is delivered.
- o OFFCLQ: Checks for messages queued for delivery to off-cluster APs. If the destination APC is now available, the messages will be delivered.
- o CLDCHK: Reads the child table for reserved entries that have not been confirmed via unsolicited init ACKs. Any entry reserved for more than the allotted time will be deleted.

3.2.3.3 Guaranteed Delivery Processing

When an application submits a Guaranteed Delivery (GD) message to the NTM by calling GDSSEND, a GD message (category A) is formatted. When the source MPU receives this message, it processes the message and logs it in the GD log (PRCONC calls GDMSGs) before returning an acknowledgement that contains the message serial number to the sending AP. The MPU continues processing this GD message by keeping track of the state of the message in the GD log. Before an MPU considers its responsibility for the GD message complete, it must receive an acknowledgement from the destination MPU (message type "GX") and return to it a MPU GO-ACKBACK (message type, "GY"). This protocol supports a hand-off mechanism that allows recovery based on the states of the messages in the GD log to take place.

When an MPU (non-comm) receives a GD message, it calls GDMSGs from MNGMSG to log the incoming GD message and sends a "GX" type ACK to the sending MPU before handing the message off to MNGPRC for the required process management and message delivery to the destination AP (via SENDAP).

The NTM internal GD protocol messages are sent as system messages (category "C" messages) and processed by SYSCOM. The GD destination AP must generate a GD ACK (by calling GDACK) when it has completed its GD processing. This GD ACK message is given a category of C, for system message, and processed by SYSCOM.

The following is the GD protocol:

1. AP calls GDSSEND service to request the delivery of a GD message.
2. GD message sent to local MPU.

3. The local MPU processed the GD request and, if accepted, logs the GD message in the GD log file and makes a short status entry in the GD table (4).
5. The local MPU sends an acknowledgement to the AP (received by GDSEND).
6. GDSEND returns control to the requesting AP with status and GD message serial number.
7. The GD message is delivered to the destination MPU (when it is available) and (8) the receiving MPU logs the message.
9. The receiving MPU sends an MPU GDACK to the sending MPU (type "GX").
10. The source MPU sends an ACKBACK (type "GY") to the destination MPU; signalling its handoff of the GD message and (11) deletes the GD log record from its file.
12. The destination MPU delivers the GD message to the destination AP (when it is available).
13. The destination AP sends a GD acknowledgement (calls GDACK) when it has completed its GD processing.
14. On receipt of the GDACK from the destination AP, the destination MPU removes its record of the GD message and (15) forwards the GD ACK to the source MPU.
16. The source MPU, on receipt of the AP's GD ACK, removes the status entry for the GD record from its GD status table.

3.2.4 Shut-Down Phase of the MPU

During the shut-down phase of the Message Processing Unit, processing continues in a mode similar to the MPU's Operations phase. PRINPT continues to read messages from the mailboxes (although there are no waits for mailbox events in the shut-down phase). Routines that process AP Status messages and system commands may still be called. The time-checking routine is not used during shut-down. In addition to the message processing routines described above, there are certain routines that are called only during the shut-down phase. These are:

- o SDMODE: Called by PRINPT to process messages after the shutdown phase has begun. This routine screens the incoming messages for AP Status Messages and System Commands. All other messages may be logged but are not processed.
- o SHTAPC: Called by SYSCOM to shut down all APs on the cluster.

- o SFTSD: Called by SHTAPC to shut down on-cluster APs having the internal logic to shut themselves down. (APs not having internal shut-down logic are aborted in SHTAPC.)

Once the shut-down of all the APs on the cluster is complete and all messages have been processed, the Message Processing Unit calls its final program, TRMAPC. TRMAPC deletes the mailboxes, writes the initialization data to the start-up file and informs the Monitor AP that the MPU is terminating.

3.2.5 Error Handling by the Message Processing Unit

Most errors occurring within the MPU will be sent to the Monitor AP via the Monitor MPU where they will be displayed on the operator's console. These messages show which MPU the message came from (the cluster on which error was detected) and what type of error has occurred (error codes are defined in the Operator's Manual [3] and the IISS Programmer's Guide [4]). The routines used to send these messages to the Monitor AP are SNDMTR and SNDMON. In most cases, SNDMON is used to send the message via the Monitor's MPU; however, during start-up on the Monitor cluster, SNDMTR is used to send the message directly to the Monitor AP's mailbox.

Error messages of interest to the message source AP are also sent. In some cases, these messages will be interpreted by the AP Interface. In the case of a "SIGERR" message, the AP will handle it itself.

3.2.6 Debugging the Message Processing Unit

During the start-up of an MPU, messages may time out if an error occurs. When debugging it may be useful to raise and lower the timeout time as needed. The timeout value during start-up is set in the sysgen file.

During regular input processing the time checking routine is invoked upon the timer, and again, it may be useful to raise or lower the timer to the debugging needs. This time value is also set in the sysgen file. Manipulation of the values in the sysgen file is discussed in Section 3.1 of this document.

3.3 NTM Services

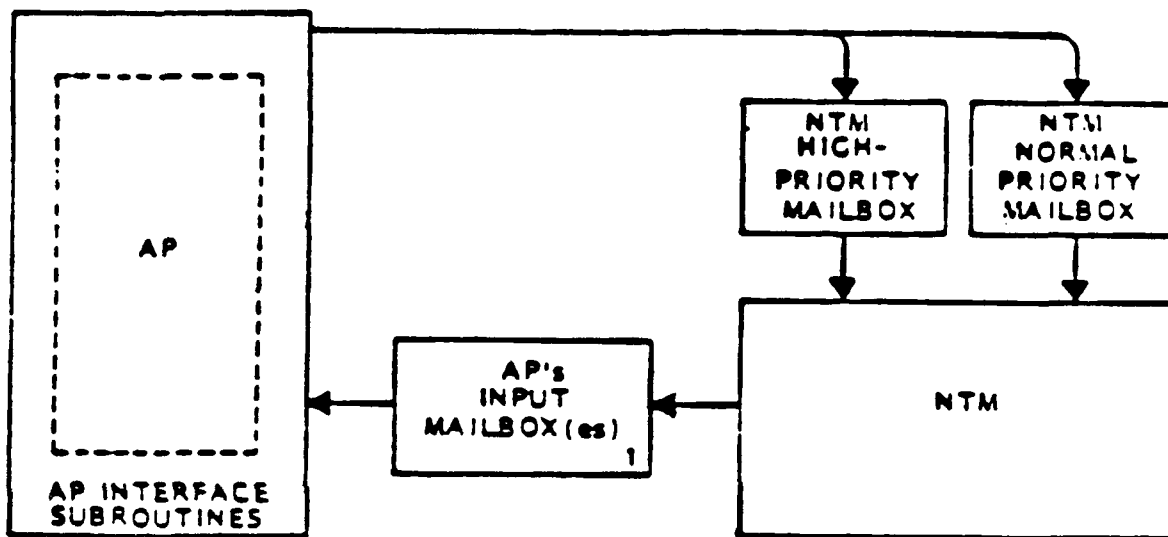
The NTM Services are a set of COBOL and C routines that are used by IISS Application Programs (APs) for communicating with other IISS APs and the NTM. The NTM Service routines are called by the APs with COBOL CALL Statements that are detailed in the IISS Programmer's Guide [4]. These Services provide the interface between an AP and the NTM by establishing, servicing, and disconnecting NTM mailbox connections for the AP (Figure 3-2). These mailbox service functions provide a logical organization of the NTM Communication Services into the Initiation, Communication and Termination classes listed in

Figure 3-4. Another category of the NTM Services contains status or information request services. Currently, there are three Status Services available, GETUSR, CHGROL, and WTHST.

The NTM Services use the Interprocess Communication Primitives (IPCs) to implement the AP's mailbox communication requests. Hence, an AP's code must be bound or linked with both the NTM Services and the IPCs. If an AP uses any CDMP (Neutral Data Manipulation (NDML) Language) or UI (Forms Processor (FP) or Virtual Terminal Interface (VTI)) Services, these CDMP and UI routines must also be bound with the AP to give the AP's executable unit (Figure 3-5).

The following sections describe the processing of the individual services within each of the four service classes initiation, communication, termination, and status.

Section 3.3.5 describes the error handling implementation in the Service routines. An interpretation guide of the error diagnostics for service debugging is also provided.



- o In implementation, APs that can process NTM commands (i.e. shutdown) will have two input mailboxes. The second input mailbox will be used to receive these high priority unsolicited messages from the NTM.

Figure 3-3. AP Mailboxes

INITIALIZATION

INITAL
INITEX*
INICOM**
LOGON*
GDSEND
QSEND

COMMUNICATION

NSEND
ISEND
RCV
CHKMSG

TERMINATION

TRMNAT
LOGOFF*
TRMNAX**

* These services are used by the UI APs only
** INICOM and TRMNAX are used by the COMM APs only

Figure 3-4. NTM Services

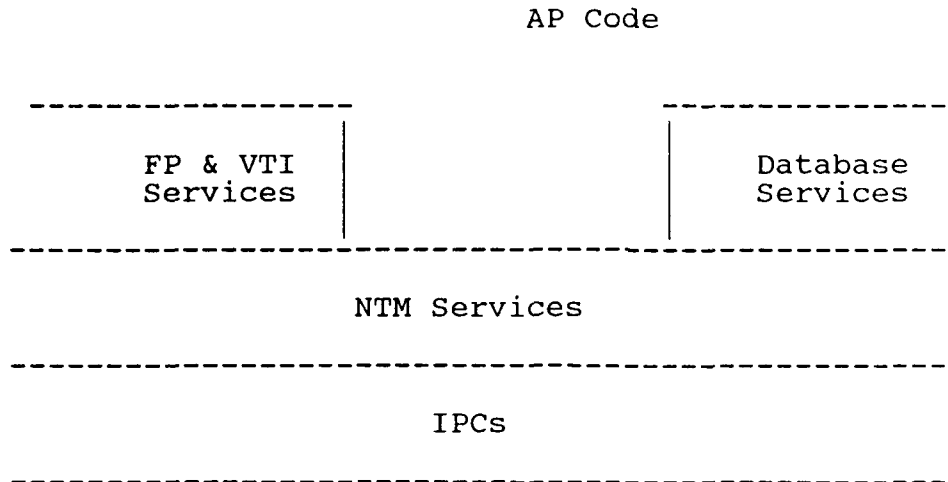


Figure 3-5. IISS AP Code Unit

3.3.1 NTM Initiation Services

The NTM Initiation Services provide three slightly different initiation routines (INITAL, INITEX, and INICOM) that support variations in the initiation requirements between the generic IISS APs, the UI AP, and the COMM AP. These initiation routines and a description of their use are in Figure 3-6.

LOGON is also a service in the initiation category. It is used by the UI to send IISS logon information to the NTM's Monitor. The Monitor keeps this logon data stored in a table to support user status requests from APs (GETUSR) and the IISS Operator. This table is the LOGTBL dataset.

The IISS architecture calls for an AP to be able to send a message (the "I'm Alive" message) to its local MPU before it receives any messages from the MPU. This requires the NTM Services to know the local MPU's mailbox names at initiation time. These MPU mailbox names are derived from the cluster's (APC) name. For example, the UIV cluster's MPU mailbox names are XUIVC and XUIVH, which signify, respectively, the MPU's cold and hot mailbox names, where "X" is the IISS instance in which the MPU is running.

The MPU's mailbox name is obtained by calling the system primitive "SGTPNM", which obtains the parent process name. This routine is used in the INITAL application service. The MPU mailbox name is "hard-coded" in the INITEX application service and has been implemented by having an include file, APCNM1, that contains the MPU name necessary to form the local mailbox name. Hence, each MPU used for support of external connections requires a separately compiled copy of the initiation routine "INITEX2" that has the correct MPU name in the APCNM1.INC file. For example, APCNM1 for the UIV cluster consists of the following declarations:

```
*      THE HARD-CODED APC-NAME....FOR EACH APC.....MUST
*      HAVE A VALUE.
01 APCNAME          PIC X(3)      VALUE "UIV".
01 UI               PIC X(3)      VALUE "UIV".
01 COMM             PIC X(3)      VALUE "COV".
01 MON-PRIME        PIC X(10)     VALUE "NTMONITV".
01 MON-PRIME-APC    PIC X(3)      VALUE "MRV".
```

This "hard-coded" requirement exists only for those APs that use the INITEX and INICOM NTM services. The mailbox name of the MPU for those APs that use INITAL is determined during INITAL processing.

<u>Initiation Service</u>	<u>User</u>	<u>Call Statement</u>	<u>Function</u>
INITAL	ALL IISS APs other than COMM and UI RET-CODE	CALL "INITAL" USING BUFFER, BUFFER-SIZE, SYSTEM-STATE mailboxes	1. Initiates NTM Data Structures 2. Creates AP's 3. Sends "I'm Alive" message to NTM 4. Receives System State Information 5. Returns System State Information
INITEX	UI BUFFER, RET-CODE.	CALL "INITEX" USING of INITIAL. BUFFER-SIZE, RET-CODE.	1. Functions 1-4 (No System State information is returned to the AP). Performs an "external" connection to the NTM (the NTM does not create the UI process).
INICOM	COMM COMM-RCV-EVENT-BLOCK, INPUT-MBX-NAME, APC-HOT-MBX-NAME, APC-COLD-MBX-NAME, RET-STATUS.	CALL "INICOM" USING of INITIAL.	1. Functions 2-3 2. Returns the mailbox names to COMM for its use in NTM communication.

Figure 3-6. INITAL, INITEX, and INICOM

3.3.1.1 INITAL

The NTM initiation service for generic APs is INITIAL. The service has four major functions: creating the AP's input mailbox(es), initializing the global data structures that are managed by the NTM services for the AP, informing the local NTM component that the AP has completed the initiation procedure (by sending an "AL" "I'm Alive" message to the MPU), and receiving system-state information from the NTM for the AP (by receiving an "SS" "System State" message from the MPU). The implementations of these four functions are described in the following paragraphs.

3.3.1.1.1 Creating the AP's Input Mailboxes

INITAL creates the AP's input mailboxes by:

1. Determining the AP's process name with a call to GETNAM. The AP's thirteen character process name (1 character IISS instance letter, 2 character directory id, 8 character AP name, and 2 character instance identifier) returned by GETNAM is the one used by the MPU to create the process.
2. Using the process name determined in Step 1 as the base of the AP's mailbox names--a C, H or A is appended to this base to form the COLD, HOT and ACK mailbox names.
3. Calling the IPC primitive, CRTMBX, to create the AP's cold and ACK mailboxes. The creation of mailboxes is dependent on the number of mailboxes the given AP supports. This information is obtained from the AP characteristics table and passed to the AP Interface in the System State Message. In order to receive the system state message, the COLD and ACK mailboxes will always be created. Depending upon the actual number of mailboxes supported by the AP, the AP interface will either create a HOT mailbox, delete the COLD and ACK mailboxes, or leave things as they stand.

3.3.1.1.2 Initializing AP Global Data Structures

The Services maintain a global data section consisting of three include files to provide message integrity for the AP. Three classes of data are maintained in this section:

- o APC and AP names used in message headers and mailbox names
- o Buffers for saving AP messages, accepted headers, and message pairing information
- o Flags that indicate state information

During initialization, the buffers, HEADER-BUFFER, RESPONSE-TABLE, and ROUTING-TABLE are cleared. Their respective pointers, NO-IN-HDR-BUFFER, NO-IN-RESPONSE-TABLE, and NO-IN-ROUTING-TABLE are set to zero.

Limits for the global buffers and other data that are considered as "local" service data are kept in APILCL.INC.

3.3.1.1.3 Informing the NTM That the AP Is Alive

After the AP's COLD and ACK mailboxes have been successfully created, an "I'm Alive" (AL) message is formatted (ALMSGH.INC) and sent to the local MPU. If the MPU's mailbox is full, the send is retried a set number of times before returning an unsuccessful status to the user.

3.3.1.1.4 Receiving System-State Information from the NTM

If the "I'm Alive" message was successfully sent to the MPU, INITAL issues a "RCVMSG", a "SETTIM" to set a wait time for the System State (SS) message, and then a "WAIT02". If the event returned on the "WAIT02" is a timer event, an unsuccessful INITAL return is given to the AP. On a mailbox event, the System State data is obtained and saved in the SYSTEM-STATE-BLOCK in the global data section (APINME.INC). If needed, mailbox processing is completed after which a successful INITAL return is given to the AP, along with the System-State Value.

3.3.1.2 INITEX

INITEX is the initiation service that is used by the User Interface (UI) to perform an external connection to the NTM. An external connection, where a running process attaches to the NTM, is necessary because of the UI terminal attachment design. The only logic differences between INITAL and INITEX are the following:

- o INITEX sends an "External Connection" (EX) request message to the MPU. The MPU will then return the "EX" message to the INITEX service with new mailbox names to be used. INITEX will then delete its original mailboxes and create new ones for use by the MPU.
- o INITEX does not return the System State value to the UI. This is because the UI does not need recovery or special restart data.
- o Since the UI must process high-priority messages from the NTM, the UI APs require two input mailboxes. Hence, there is no need for INITEX to determine the number of input mailboxes for the UI. NUM-OF-MBXS has a value of "2".

3.3.1.3 INICOM

INICOM is the initiation service that is used by the COMM APs. Because COMM directly handles the mailbox sends and receives between it and the NTM, there are some differences in the parameter requirements between INITAL and INICOM.

INICOM's functions consist of:

- o Creating COMM's input mailbox.
- o Sending the COMM's "I'm Alive" message to the local MPU.
- o Returning the mailbox names and initiation status (RET-STATUS) to the COMM AP.

NOTE: Because of the IPC requirement for the event-block address used in the CRTMBX call to be the one used in all RCVMSG on that mailbox, the COMM AP must pass the input mailbox event block address to INICOM.

3.3.1.4 LOGON

LOGON is the service used by the UI to transmit IISS terminal LOGON information to the NTM. After an IISS users logon has been accepted by the UI, it calls LOGON with the statement:

```
Call "LOGON" USING  TERMINAL-ID,  
                    USER-NAME,  
                    ROLE-NAME,  
                    SESSION-START-TIME,  
                    CHAN-RANGE-START,  
                    CHAN-RANGE-END,  
                    RET-CODE.
```

The LOGON Service formats an NTM message that contains the data received in the CALL arguments and sends the message to the MONITOR on the primary Host via the local MPU. The message structure is in LGNMSG.INC. Control is returned to the calling AP upon receipt of a message from the Monitor AP containing the status of the Logon entry. If the entry could not be made, the Logon will be disabled.

3.3.2 NTM Communication Services

The NTM Communication Services are those that process the AP's "mail" requests. There are five basic Communication Services available: NSEND, ISEND, QSEND, RCV, and CHKMSG. In addition, TSTMOD and SIGERR are available to enable the send and receipt of AP error messages. The AP uses the NSEND service to send regular mail or data to other IISS APs and uses the ISEND service to send a special request to start a new instance of an AP. A queue-server AP, however, would use the QSEND service instead in order to send the message directly back to the AP from which the requesting message was received and to terminate the connection with that AP. To receive messages that have been delivered to the AP, the AP uses the RCV service. CHKMSG is the service that allows the AP to check to see whether it has received any mail. If the AP wishes to receive AP error messages, it will call the TSTMOD service to set that condition. AP's wanting to send error messages will do so by calling the SIGERR service.

The logic of these Communication Services is described in the following paragraphs.

3.3.2.1 NSEND

The primary function of the NTM Service, NSEND, is to packetize the data supplied by the calling AP in an NTM message and send it, via the local MPU, to the requested destination. In order to perform this function, several subtasks must be done. These tasks are:

- o Determine the number of message packets required (user data length divided by the maximum message data length - rounded up).
- o Formats the message header.
- o Send the message(s) to the MPU (SNDMSG).
- o On messages that require an acknowledgment from the MPU, the timer ("SETTIM") is set, a receive on the ACK mailbox ("RCVMSG") is issued, followed by a wait call for a mailbox or timer event ("WAIT02"). On a mailbox event, the message ("GETMSG") is retrieved, and then NSEND determines whether it is an acknowledgment (Type "MA") or NACK (negative acknowledgment). If it is an acknowledgment, processing proceeds by either returning a successful status to the AP or returning to the message formatting task to send the remaining packets for the user's data. A NACK causes a "message no-accept" status to be returned to the user. ACK and NACK messages are sent to the AP's ACK mailbox as the AP will only read the ACK mailbox at this time, there is no possibility of a buffer overflow prior to the receipt of the ACK or NACK Messages that may arrive at this time are sent to the AP's hot or cold mailbox (or queued) for later processing. If an expected acknowledgment is not received within a specified time frame (RCV-WAIT), the AP is returned an unsuccessful NSEND status.

There are several bookkeeping jobs that NSEND must perform in order to correctly process the AP's request. These are the following:

- o Pairing Logic Support:

The NTM Services must determine the correct message category for the message header. It does this according to the following logic.
- o If the time-out value is set by the AP in the NSEND calling arguments (non-zero), a paired message (Category "B") is indicated. If the paired request requires continuation messages, only the first message has the category field set to "B". All the following continuation packets for this request have their category field set to "unsolicited" (E).

- o If the message is not a paired request, NSEND must determine whether the message is a response to a paired message. It checks the services global table, RESPONSES-WAITING, to determine if the message is a response to the specified destination on the channel indicted in NSEND's calling arguments. The RESPONSES-WAITING table is filled by the RCV service when it receives a message for the AP that requires a response. If a response entry is found by NSEND, the entry is deleted from the table and the message category is marked as a response (category F). For response messages with continuation packets, only the first message has its category field set to response. All others have their category set to "unsolicited" (E).
- o If the message is not a paired request or a response, the category field is set to "unsolicited" (E).
- o Header Formatting Optimization

To optimize the number of message acknowledgments required for AP messages, a HEADER-BUFFER that contains headers already completed and accepted by the local MPU is kept. On initiation, NSEND checks this buffer to determine whether there is an accepted header for the specified destination. If one exists, it is used to format the messages related to this current NSEND call. A header field that signal "old header" to the MPU is set. This disables the acknowledgment function in the MPU and NSEND.

For continuation messages, only the first message can possibly require a complete new header processing (FORMAT-NEW-MESSAGE-HEADER paragraph) and acknowledgment. All succeeding messages are marked as "old headers" and no acknowledgments by the MPU are required.

3.3.2.2 ISEND

ISEND is special purpose "send message" service that is used when an AP specifically wants to initiate a new instance of an AP. It differs from NSEND in one area:

- o ISEND performs the same categorization logic as NSEND but will assign category "H" (Specific Initiation) or category "J" (Specific Initiation Response Required) based upon the timeout value in the calling arguments.

3.3.2.3 QSEND

QSEND is a special purpose "send message" service that is used by a queue server AP to terminate a connection with a parent AP.

A queue server is a special type of application which performs processing on behalf of multiple parent applications. The parent application in the context of AP-to-message dialogue is the AP which initiates the dialogue by sending the first

message. In non-queue server APs, the first message will ordinarily cause initiation of the "child" AP, hence the terms "parent" and "child". Although the queue server functions as a child, it may have multiple simultaneous "parents". Thus, the first dialogue message from a specific parent will result in a new connection, and only the first parent can cause initiation of the queue-server AP. Subsequent parents form new connections. These messages arise from utilization of the NSEND (or ISEND) service. NSEND, ISEND, and RCV cooperate to manage queue server connections via a connection table maintained for each queue server.

When a specific dialogue is ended, a non-queue server AP would simply terminate. The queue-server AP, however, must remain active to process work for other parents. Consequently, it must have a way to end connections in order to allow new connections to be formed. This is the function of the QSEND service. The QSEND service is used only by queue servers to terminate a connection with a parent. (For more information on queue servers see section 6 of the NTM Programmer's Manual, reference 4).

3.3.2.4 RCV

RCV is the NTM Service used by all IISS APs to get message data that has been sent to them by other APs and the NTM. The RCV CALL has the following format:

```
CALL "RCV" USING LOGICAL-CHANNEL,  
                  WAIT-FLAG,  
                  MSG-SOURCE,  
                  MESSAGE-TYPE,  
                  DATA-LENGTH,  
                  USER-DATA,  
                  ACCEPT-STATUS,  
                  MSG-SERIAL-NUMBER.
```

The input arguments, LOGICAL-CHANNEL, WAIT-FLAG, and MSG-SOURCE determine the type of receive requested and actually serve to partition the RCV code. LOGICAL-CHANNEL and MSG-SOURCE determine the type of message requested by the AP. On entry into RCV, a variable, MATCH-CRITERION, is set to ANY-M (if both LOGICAL-CHANNEL and MSG-SOURCE are not set), S-MATCH (if MSG-SOURCE only is set), C-MATCH (if LOGICAL-CHANNEL only is set), or S-C-MATCH (if both are set). If WAIT-FLAG is set, the RCV-AND-WAIT paragraphs are executed. Otherwise, the RCV-AND-NOWAIT paragraphs are executed.

The logic of RCV is the following:

- o Determine the message MATCH-CRITERION described above.
- o Check the Service global buffer, MSG-RCV-BUFFER, for AP messages that meet the MATCH-CRITERION. If there is a message in the buffer that meets the MATCH-CRITERION, the correct arguments are returned to the calling AP. The

value of ACCEPT-STATUS is determined by the message category and the message-type (if the source of the received message is the NTM).

- o If there are no messages in the buffer that meet the MATCH-CRITERION, perform the correct RCV paragraph (WAIT or NO-WAIT).
- o The number of RCVs to be issued is based on a combination of the calling arguments and the number of mailboxes the AP supports. Therefore, if the AP supports two mailboxes and is looking for any message, a RCV will be set on both the hot and cold mailboxes. Figure 3-7 provides a matrix of the combinations to show the number of RCVs issued under the possible conditions.

AP's #MBXS \	RECEIVE CALLING ARGUMENT (NTMPU)	HOT MES- SAGE ONLY S-MATCH	S-C MATCH	C-MATCH	MATCH	ANY
0		Error	Error	Error	Error	Error
1		Error Cold	RCV on Cold	RCV on Cold	RCV on Cold	RCV on
2	Hot only	RCV on Cold only	RCV on Cold only	RCV on Cold	RCV on Hot or Cold	RCV on Hot or

Figure 3-7. RCV Condition Matrix

- o On a RCV-AND-WAIT, the correct number of "RCVs" are issued, and the "WAIT01" or "WAIT02" is called to set a wait for a mailbox event. On an event, the received message is checked (CHECK-FOR-REQUESTED-MESSAGE) to determine whether it meets the MATCH-CRITERION. If the message received is not the requested message, the message is stored in MSG-RCV-BUFFER (paragraph PUT-IN-BUFFER). This step is repeated until a match is found and all packets belonging to the message have been received. Control is then returned to the calling program.
- o On a RCV-AND-NOWAIT, "RCVMSG" and then "GETMSG" are issued again, based on the conditions for hot, cold or both. If "no message" is indicated on the "GETMSG" return, ACCEPT-STATUS is set to indicate "No-message" and control is returned to the calling AP. Otherwise, the received message is processed (CHECK-FOR-REQUESTED-MESSAGE) in the same manner described in the RCV-AND-WAIT paragraph above. The only exception is that the receiving process is not repeated unless the message indicates a continuation. Then, the

process is repeated until all the packets are received. If the received message does not meet the match criterion, ACCEPT-STATUS is set to indicate "No-message".

- o When a message is about to be returned to the caller, the return arguments and any necessary pairing arguments are set. Routing information is also saved because if the calling AP is a queue-server, it would be needed for a QSEND return message. If the message is of Category B or J (Response-required), the response information is stored in the RESPONSE-TABLE (paragraph STORE-RESPONSE). The routing table is searched (ROUTE-TABLE-TASK) to determine whether the message source has an entry. If one already exists, nothing is done. If one does not exist, an entry is made in ROUTING-TABLE for the Source AP. This entry consists of the Source AP name, its APC name, and the Source AP's instance number. It will be removed when QSEND is called to send a return message. If the ROUTING-TABLE fills up, which would occur if we are not a queue-server, we simply do not add any more entries and continue since ROUTING-TABLE entries were obviously not being used anyway. If the received message is a guaranteed delivery message (Category A) then the APC name of the source and the message serial number are stored in the local buffer area GD-TABLE.

3.3.2.5 CHKMSG

CHKMSG is the NTM Service that determines whether there are messages that have arrived for the calling AP. Like RCV, the calling AP may request a check for any message, a message from a specific source on any channel, one from any source but on a specific channel, or one from a specific source on a specific channel. All that is returned to the caller is the ACCEPT-STATUS that indicates whether or not a message of the indicated type is available; the message itself must be retrieved by calling RCV. The CHKMSG call has the following format:

Call "CHKMSG" Using Logical-Channel,
Msg-Source,
Ret-Code.

Like RCV, the buffer, RCV-MSG-BUFFER, is first checked to determine if there are any messages that match the request criterion of the call. If a message is found a successful status is returned to the calling AP.

If there is no message in the receive buffer that meets the request criterion, "RCVMSG" and then "GETMSG", with no WAIT, are called. The hot mailbox is checked only if the AP requested messages from the NTM (specifying MSG-SOURCE as "NTMPU"). The AP's mailboxes are read until a message that meets the request criterion is found, the mailbox is empty, or the global buffer RCV-MSG-BUFFER and BUFFER-OVERFLOW-AREA are filled. The ACCEPT-STATUS return to the calling AP will indicate the CHKMSG status--either "message found", "no message", or "buffer overflow".

Note that no AP messages are lost on a "buffer-overflow" return. The last message is stored in the overflow area and the condition will be corrected when the AP issues a RCV.

3.3.2.6 TSTMOD

TSTMOD is the service that sets the condition for an AP's receipt of error messages from its child APs. The AP may choose one of three options: Receive all error messages; receive only fatal error messages; or receive no error messages. When called, the TSTMOD service will set the appropriate flag. Subsequent error messages are screened in accordance with the criteria requested by the AP and deliver only those messages that meet the criteria. The TSTMOD call has the following format:

Call "TSTMOD" using Test-Status,
Ret-Code.

3.3.2.7 SIGERR

SIGERR allows the calling AP to describe an error condition to its parent AP. SIGERR will format a message (type "SE") using the values from the calling arguments. The formatted message is sent to both the parent AP and to the Monitor AP. The message will be delivered to the parent AP if it meets the current criteria set by TSTMOD. The message will be displayed to the IISS operator if SIGERR is enabled (operator command SE). The SIGERR call has the following format:

Call "SIGERR" using ERROR-CODE,
SEVERITY-LEVEL,
ERROR-MESSAGE,
RET-CODE.

3.3.3 NTM Status Messages

There are three NTM Status Messages currently available. These are CHGROL, GETUSR and WTHST. CHGROL is used by the UI to inform the NTM that an IISS user who is currently logged on has changed roles. GETUSR is called by the AP to determine information about its original source. WTHST is used to find the name of the Host Machine upon which a given AP currently resides. These services are described in the following paragraphs.

3.3.3.1 CHGROL

Like LOGON and LOGOFF, CHGROL is a service used only by the UI to inform the NTM of IISS User logon information. The call format is:

Call "CHGROL" USING TERMINAL-ID,
USER-NAME,
ROLE-NAME,
RET-CODE.

The logic is identical to that in LOGON and LOGOFF. It involves a simple message formatting procedure that fills in the message structure for the CHGROL message contained in CHGMSG.INC. After the message formatting procedure is complete, CHGROL sends the message to the monitor on the primary host via the local NTM.

3.3.3.2 GETUSR

GETUSR is the NTM Service called by an AP when it needs to know specific data about it's original source. This need may arise when an AP is part of a chain and wants to send a message to the originator of the chain. It can also occur when an AP, as in the case of MCMM, needs to know logon data regarding the user who started the chain for database access control.

The GETUSR CALL is:

```
Call "GETUSR" USING R-APNAME,  
LOGICAL-CHANNEL,  
USER-NAME,  
ROLE-NAME,  
TERM-ID,  
RET-CODE.
```

The global section of the Services contains the original source AP data. The user's logon data is kept in the Logon table which is maintained by the Monitor AP. This logon data is only required if the original source is the UI (a user at a terminal serviced by the UI). The logic of GETUSR then:

- o First determines whether the original source of the AP is the UI. If not, the R-AP-NAME and LOGICAL-CHANNEL return arguments are set with the correct values and the user data argument are set to blanks. The return values are then given to the calling AP.
- o If the original source AP is the UI, a message requesting the user logon information is sent to the Monitor AP. The service NSEND is used here with the message type set to "UX" to indicate the GETUSR data request.

Following a successful NSEND, a RCV (with wait) is issued for the response message from the Monitor. On a successful return from RCV, the user data is returned to the calling AP.

3.3.3.3 WTHST

WTHST is used by any AP wishing to know the host location of itself or any other AP in the IISS. The call format is:

```
Call "WTHST" USING      AP-NAME,  
AP-HOST-NAME,  
RETURN-STATUS.
```

If the AP-NAME argument is blank, the service will assume that the request is for the calling AP's location. The service will format a message to the local MPU requesting the host name of the given AP. The local MPU will process the request by searching the tables. If the given AP is found, it's host name is returned. If it is not found, a status message to that effect will be returned. The WHTHST service will interpret the message from the MPU and return the information to the calling AP.

As in the GETUSR service, WHTHST uses the "NSEND" and "RCV" services for its dialog with the MPU.

3.3.4 NTM Termination Services

The NTM Termination Services are used to signal the end of an IISS user session to the NTM (LOGOFF) or the end of an AP's IISS processing (TRMNAT). These two services, LOGOFF and TRMNAT, are described in the following paragraphs.

3.3.4.1 LOGOFF

LOGOFF, like LOGON and CHGROL, is used by the UI to inform the NTM of an IISS user's logon status. When an IISS user terminal logoff has been received by the UI, it informs the NTM of the event with a call to LOGOFF. The LOGOFF call syntax is

```
Call "LOGOFF" USING TERMINAL-ID,  
USER-NAME,  
ROLE-NAME,  
RET-CODE.
```

The LOGOFF routine uses the three input argument values to format the LOGOFF message (LGFMSG.INC) which is then sent to the primary Monitor (currently the VAX) via the local MPU.

3.3.4.2 TRMNAT

All IISS APs use the NTM service, TRMNAT, to terminate their connection to the IISS. The TRMNAT call syntax is:

```
Call "TRMNAT" USING TERMINATION-STATUS.
```

TRMNAT has the following three major tasks.

- o Informs the local MPU that the AP is terminating. TRMNAT does this by sending an "I'm Dying" message to the MPU. The format of this message is in the include file ADMSGH.INC.
- o Deletes the AP's mailboxes by calling the IPC "DELMBOX" for each AP mailbox.
- o Stops the AP's processing with a call to the IPC, "ENDRUN", which terminates the AP.

3.4 NTM Internal Table Access and Control

The NTM tables provide both static configuration and dynamic operating data. This data is continually accessed by the NTM components for routing, processing, and various bookkeeping functions. The table functionality and content is discussed in detail in the NTM Development Specification [1]. Structure Charts and Module descriptions of the table routines are available in the NTM Product Specification, Vol. 3 [2].

The table handling routines consist of two generic types: initialization and access. Within these types, the static and dynamic tables are handled differently. The table initialization routines for static tables are called during start-up processing to load static data into memory. This static data provides the basic system configuration information such as, the location of APs within the IISS, the initial status of the APCs and hosts, and processing requirements of the various message categories. When called, the initialization routines will open the appropriate file, read the existing records into memory, and close the file. Each static table initialization routine follows the same logic; the only difference between them is the file which is accessed.

The dynamic table initialization routines serve to clear out space in memory sufficient to hold the number of records mandated by the occurs clause in the table description.

The table access routines provide the functionality for reading, writing, modifying, and deleting records. The calling arguments specify the table to be accessed, the function to be performed on the table, a search field and value (if needed), a search start flag, an index (if known), and a record buffer. Based on the given parameters, the routine will perform the requested function. Each routine follows the same basic logic, again with different handling for static and dynamic tables. As with the initialization routines, the major difference is the table which is accessed.

The following is a list of the functions provided by the "Table" access and control routines:

- o RS - Read Search
- o RA - Read All
- o WT - Write New
- o MD - Modify Old
- o DI - Delete Index

The Dynamic Table routines perform all of the above functionality. They will attempt to use in-core memory by re-using space left by deleted entries. Where all the in-core space is occupied, these routines will access files. These files

are organized in Index-Sequential format with the local APC name as part of the primary key. The exception to this is the Logon Table which is organized sequentially (as it is accessed only by the monitor AP).

The static tables are also resident in memory. If needed, file access calls will check records kept on disk. The static table routines contain all the functionality described above but the write and delete functions are not currently used.

Table initialization routines are named "xxxINI" where "xxx" is the abbreviation for the specific table being initialized (such as "CAT" for the message category table - "CATINI"). Using the same convention, the access routines are named "xxxTBL" where "xxx" is the abbreviation for the table being accessed (message category table - "CATTBL"). The static configuration data is contained in files which follow the naming convention "xxxTBL.DAT" where "xxx" is the abbreviation for the specific table to be initialized (such as "CAT" for the message category table "CATTBL.DAT"). Each table is described in an include file named according to the convention "TBLXXX. Inc." where xxx is the table name.

SECTION 4

DATA STRUCTURES

4.1 Sysgen Data Processing

A portion of the information required for system initialization is contained in one file that is read by both the Monitor AP and each MPU at startup. This file contains those data items that (a) are subject to change and (b) should not be hard coded.

The data file that is read at startup is called SYSGEN.DAT. The structure of the data items in this file are shown below. The value of these items may be modified by running the routine MPUGEN. This routine provides the user with the main options shown below.

SYSGEN OPTIONS AVAILABLE:

1. CREATE DEFAULT SYSGEN FILE
2. CREATE NEW IISS SYSGEN DATA RECORD
3. MODIFY EXISTING IISS SYSGEN DATA RECORD
4. CHANGE IISS INSTANCE
5. MODIFY EXISTING APC DATA RECORD
6. DELETE EXISTING APC DATA RECORD
7. ADD NEW APC DATA RECORD
8. QUIT

PLEASE ENTER NUMBER OF YOUR CHOICE

The offered options are described below

o CREATE DEFAULT SYSGEN FILE

This option uses the default initial sysgen parameters to create an initial SYSGEN.DAT file with all message serial number seeds at 0. If a new default value is defined, MPUGEN must be modified to support it.

01 REC-1.		
05 HEADER-REC-1	PIC X(15)	VALUE "MTRSYGENLOCALS"
05 IISS-..	PIC X	VALUE "A".
05 INIT-HOST-NAME	PIC X(3)	VALUE "VAX".
05 INIT-APC-NAME	PIC X(3)	VALUE "MRV".
05 INIT-MPU-NAME	PIC X(10)	VALUE "NTNTMRVMPU".
05 INIT-COM-APC	PIC X(3)	VALUE "COV".
05 INIT-UI-APC	PIC X(3)	VALUE "UIV".
05 INIT-BASEPRI	PIC X(2)	VALUE "00".
05 INIT-PROC-TYPE	PIC X	VALUE "M".
05 INIT-MODULE-NAME	PIC X(10)	VALUE "NTNTMONITV".
05 INIT-TIMER-DELAY	PIC X(6)	VALUE "000500".
05 INIT-TABLE-STATUS	PIC X	VALUE "0".
05 INIT-LINK-COUNT	PIC X	VALUE "1".
05 FILLER-REC-1	PIC X(26)	VALUE SPACES.
01 REC-2		
05 HEADER-REC-2	PIC X(15)	VALUE "MTRSYGENHSTLNK"
05 LINKID	PIC X(2)	VALUE "VI".
05 COMMAP	PIC X(10)	VALUE "COCOMVICOM".
05 HSTNAM	PIC X(3)	VALUE "IBM".
05 MTRNAM	PIC X(10)	VALUE "NTNTMONITI".
05 MTRAPC	PIC X(3)	VALUE "MKI".
05 NODE	PIC X	VALUE "0".
05 PORT-NAME	PIC X(12)	VALUE "TTB3".
05 FILLER-REC-2	PIC X(29)	VALUE SPACES.

Figure 4-1. Monitor's Default Sysgen Data

01	IISS-SYSGEN-DATA.		
05	IISS-HEADER	PIC X(9)	VALUE "HSTSYSGEN".
05	ID-HOSTNAME	PIC X(3)	VALUE "VAX"
05	ID-LOCMTR	PIC X(10)	VALUE "NTNTMONITV"05 05
05	ID-MSTRMTR	PIC X(10)	VALUE "NTNTMONITV".
05	ID-LOCMTR-A	PIC X(3)	VALUE "MRV".
05	ID-MSTRMTR-A	PIC X(3)	VALUE "MRV".
05	ID-CDMAPC	PIC X(3)	VALUE "CDM".
05	ID-COMMAPC	PIC X(3)	VALUE "COV".
05	Q-INDICATOR	PIC X	VALUE "0".
05	MAX-NO-QUEUED	PIC X(4)	VALUE "9999".
05	MAX-LAST-KEY	PIC X(9)	VALUE "999999999'.
05	COM-LOG-INDICATOR	PIC X	VALUE "0".
05	LOCAL-MBX-SIZE	PIC X(5)	VALUE "04000".
05	ST-TIME-CHECK-INTERVAL	PIC X(6)	VALUE "000100".
05	TIME-CHECK-INTERVAL	PIC X(6)	VALUE "000100".
05	NUM-WRITE-TRIES	PIC X(4)	VALUE "0020".
05	IAT-MAX-NUM-TRIES	PIC X(4)	VALUE "0003".
05	IISS-INSTANCE	PIC X	VALUE "A".
01	MRV-SYSGEN-DATA		
05	MPU-HEADER	PIC X(9)	VALUE "MPUSYSGEN".
05	ID-APCNAME	PIC X(3)	VALUE "MRV"
05	ID-MSGSN	PIC X(7)	VALUE "0000000".
05	MPU-FILLER	PIC X(66)	VALUE SPACES.
01	CDM-SYSGEN-DATA		
05	MPU-HEADER	PIC X(9)	VALUE "MPUSYSGEN".
05	ID-APCNAME	PIC X(3)	VALUE "CDM".
05	ID-MSGSN	PIC X(7)	VALUE "0000000".
05	MPU-FILLER	PIC X(66)	VALUE SPACES.

Figure 4-2. MPU's Default Sysgen Data

01 UIV-SYSGEN-DATA		
05 MPU-HEADER	PIC X(9)	VALUE "MPUSYSGEN".
05 ID-APCNAME	PIC X(3)	VALUE "UIV".
05 ID-MSGSN	PIC X(7)	VALUE "0000000".
05 MPU-FILLER	PIC X(66)	VALUE SPACES.
01 COV-SYSGEN-DATA		
05 MPU-HEADER	PIC X(9)	VALUE "MPUSYSGEN".
05 ID-APCNAME	PIC X(3)	VALUE "COV".
05 ID-MSGSN	PIC X(7)	VALUE "0000000".
05 MPU-FILLER	PIC X(66)	VALUE SPACES.
01 T1V-SYSGEN-DATA		
05 MPU-HEADER	PIC X(9)	VALUE "MPUSYSGEN".
05 ID-APCNAME	PIC X(3)	VALUE "T1V".
05 ID-MSGSN	PIC X(7)	VALUE "0000000".
05 MPU-FILLER	PIC X(66)	VALUE SPACES.

Figure 4-2. MPU's Default Sysgen Data (Continued)

o CREATE NEW IISS SYSGEN RECORD

This option will replace the IISS Sysgen record in SYSGEN.DAT with a new record which is built by prompting the user for the new values for certain fields in the IISS sysgen record (as listed in Figure (4-3)). The IISS instance identifier is not changed in this option.

o MODIFY EXISTING IISS SYSGEN RECORD

This option will replace the IISS sysgen record in SYSGEN.DAT with a new record which is built by prompting the user for the new values for certain fields in the IISS sysgen record which the user wishes to modify. The user has the option of changing some values while leaving others as defined.

o CHANGE IISS INSTANCE

This option allows the user to change the IISS instance value in both the monitor sysgen record and the MPU sysgen record.

o MODIFY EXISTING APC DATA RECORD

This option allows the modification of the message serial number seed in the APC sysgen data record.

o DELETE EXISTING APC DATA RECORD

This option deletes the APC sysgen data record for a selected APC.

o ADD NEW APC DATA RECORD

This option allows the user to add a new APC sysgen record record to the sysgen file.

o QUIT

Writes any modified values to SYSGEN.DAT and exits MPUGEN.

Host Name

Local Monitor AP Name

Master Monitor AP Name

Local Monitor's APC Name

Master Monitor's APC Name

CDM APC Name

COMM APC Name

Queue Indicator

Maximum Number Queued

Max Last Key

COMM Log Indicator

Local Mailbox Size (sets the size of all mailboxes)

Startup Time Check Interval (Monitor AP Timer)

Timer Interval (MPU Timer)

Number of Write Tries (number of times the MPU will attempt to write to a full mailbox)

IAT Max Number of Tries (number of times an I'm alive entry will be checked before concluding that the AP did not initiate).

Figure 4-3. SYSGEN Data Items Modified or Added via MPUGEN

The use of this routine allows the IISS operator to set runtime values of the data items noted in Figure 4-3. In addition, the IISS instance ID may be modified in order to run multiple IISS images at one time. Any changes made become effective during the next execution of the start IISS command. No recompilation is necessary.

4.2 Monitor Application Process

4.2.1 Monitor AP Internal Table

The Monitor AP has a special internal table, called the Host-Link Information Table (HLT), which provides information regarding remote IISS hosts, the comm links to communicate with these remote hosts, and the remote monitor APs. This table is maintained as part of the Sysgen parameters and is read into the Monitor AP during startup. When a new host is introduced into the IISS environment, a new entry must be made in this table. This is currently accomplished by modifying the routine MPUGEN. The Host Link table entries are described in REC-2 in Figure (4-1). The INIT-LINK-COUNT Value in REC-1 will also have to be modified to reflect the correct number of links to remote hosts that must be established at startup.

4.3 Message Processing Unit

During IISS start-up the MPU uses APC Initialization data which is contained in an internal data structure called "IDATA". At runtime, the MPU uses various queues in its operation. These data structures are described below.

4.3.1 Queues

4.3.1.1 Off-Cluster Message Queue

Messages destined for an off-cluster AP that cannot be delivered immediately are written sequentially to a file. This file serves as the off-cluster message queue. Periodically, the MPU process this queue by reading and delivering each message in first-in-first-out order. This message queue is the APCQUE data set. It is an indexed-sequential dataset.

4.3.1.2 On-Cluster Message Queue

Messages destined for application processes on the cluster that cannot be delivered immediately are written sequentially to a file. This file serves as the on-cluster message queue. Periodically, the MPU processes this queue by reading and delivering each message in first-in-first-out order (for each AP). This message queue is the APQUE data set. It is an indexed sequential dataset.

4.3.2 APC Initialization Data

During the start-up of an AP cluster, the MPU reads the SYSGEN data file to obtain its local operating data. This information is then stored in the data structure IDATA. The

MPU's process name is determined by calling the NTM service, GETNAM, and is also stored in IDATA. The data structure IDATA is defined in INIDAT.INC and is passed to every MPU sub-routine

```

01 IDATA
  03 IDATA-DAT.
    05 ID-APCNAME          PIC X(3).
      88                  VAX-COMM          VALUE "COV".
      88                  IBM-COMM          VALUE "COI".
      88                  HWL-COMM          VALUE "COH".
      88                  IBM-MTR           VALUE "MRI".
      88                  VAX-MTR           VALUE "MRV".
      88                  HWL-MTR           VALUE "MRH".
      88                  TMP-UI            VALUE "UIV".
      88                  VAX-TESTAP        VALUE "T1V".
      88                  HWL-TESTAP        VALUE "T1H".
      88                  CDM-AP            VALUE "CDM".
    05 ID-MSGSN            PIC X(7).
    05 ID-HOSTNAME         PIC X(3).
    05 ID-LOCMTR           PIC X(10).
    05 ID-MSTRMTR          PIC X(10).
    05 ID-LOCMTR-APC       PIC X(3).
    05 ID-MSTRMTR-APC     PIC X(3).
    05 ID-CDMAPC           PIC X(3).
    05 ID-COMMAPC          PIC X(3).
    05 Q-INDICATOR         PIC X.
      88 ONE-MPU-ONLY      VALUE "1".
      88 DO-ALL            VALUE "0".
    05 MAX-NO-QUEUED       PIC 9(4).
    05 MAX-LAST-KEY        PIC 9(9).
    05 COM-LOG-INDICATOR   PIC X.
    05 LOCAL-MBX-SIZE      PIC 9(5).
    05 ST-TIME-CHECK-INTERVAL PIC 9(6).
    05 TIME-CHECK-INTERVAL PIC 9(6).
    05 NUM-WRITE-TRIES     PIC 9(4).
    05 IAT-MAX-NUM-TRIES   PIC 9(4).
    05 IISS-INSTANCE       PIC X.
  03 ID-MPU-PROC-NAME.
    10 MPU-PREFIX          PIC X(2).
    10 MPU-APNAME          PIC X(5).
    10 MPU-PART            PIC X(3).
    10 MPU-INST            PIC XX.

```

```
03 Q-TABLE.  
  05 MAX-IN-Q-TABLE          PIC 9(4).  
  05 NO-IN-Q-TABLE           PIC 9(4).  
  05 CLQTBLL OCCURS 10 TIMES.  
    10 CLQ-APCNAME           PIC X(3)  
    10 CLQ-NUMMSG            PIC 9(4)  
    10 CLQ-LASTKEY           PIC 9(9).  
03 ACK-FLAG                   PIC X.  
  88 YES-ACK                  VALUE "1".  
  88 NO-ACK                   VALUE "0"
```

4.4 NTM Internal Tables

During its operation, the NTM maintains seventeen internal tables (see Figure 4-4). These tables contain static data, dynamic data, or a combination of both. Figure 4-5 identifies the tables as members of these categories.

Static tables are those whose records are fixed. These records are read by the NTM components but not changed during IISS operation. These tables are initialized at start-up.

Dynamic tables are those whose records are created and used only during system operation. At start-up, these tables are initialized as empty space.

Static/Dynamic tables are those whose records include items whose values are changed during system operation. These tables are initialized at start-up with the dynamic values set for start-up conditions.

The IISSCLIB copy library contains a member for each table. These copy members specify the internal description of each table in COBOL syntax.

The tables are further defined as being global or local as follows:

- o Global tables are those that are known to all NTM components on one specific host (such as the VAX). There exists only one copy of this type of table and it is contained in the system-wide shared ("global") memory.
- o Local tables are those that are known only to a specific AP Cluster (such as the monitor MPU). Only one copy of this table will exist on any one AP cluster and it will be contained within the AP cluster's shared memory.

In either case, access to the tables is provided by the NTM table access routines, which are described in Section 2.4 of this document. These internal tables are never accessed directly, only through the use of the NTM table access routines.

Table Name -----	Table ID -----	Table Location -----
Message Category Table	CAT	Global-IISS
Authority Table	AUT	Local to APC (Based on resident AP's authority to send)
AP Information Table	API	Local to APC
AP Status Table	APS	Local to AP's AP Cluster
Child Table	CLD	Local To Parent AP's AP Cluster
AP Character- istics Table	APT	Local to AP's AP Cluster
APC Status Table	APC	Global IISS
Host Status Table	HST	Global-IISS
Message Pairing Table	MPR	Local to sending APC
Logon Table	LOG	Local to Monitor AP

Figure 4-4. NTM Tables - Overview

Table Name -----	Table ID -----	Table Location -----
Guaranteed Delivery Table	GRD	Local to sending APC
AP Operating Information	APO	Local to AP's AP Cluster
I'm Alive Table	IAT	Local to AP's AP Cluster
Link Status Table	LST	Local to Monitor AP
Authority	APC	ACT Local to Check Table
Directory Table	DIR	Global - IISS
Connection Table	NAC	Local to APC

Figure 4-4. NTM Tables - Overview (Continued)

Static Tables -----	Dynamic Tables -----	Static/Dynamic Tables -----
Message Category Table	AP Status Table	APC Status Table
Authority Table	Child Table	Host Status Table
AP Information Table	Message Pairing Table	AP Operating Information
	Logon Table	Link Status Table
AP Characteristics Table	Guaranteed Delivery Table	
Authority Check Table	I'm Alive Table	
Directory Table	Connection Table	

Figure 4-5. NTM Tables by Data "Stability"

SECTION 5

HOST SYSTEM DEPENDENT SPECIFICS

5.1 Introduction

As was stated earlier in this document, the NTM uses the services of the underlying host operating system to perform some of its functions. There are a set of NTM program modules whose implementations are host system dependent. These routines have the same high level calls on all of the IISS hosts. These routines are:

- o MPUINI The MPU initiation routine
- o SENDAP The MPU routine to handle the on-cluster message queue using variable length records.
- o OFFCLQ The MPU routine to handle the off-cluster message queue using variable length records.
- o GDMSGGS The MPU routine to handle the guaranteed delivery message queue using variable length records.
- o NQINIT The MPU routine to store messages that cannot initiate an application due to number of instances are active. This is also a variable length record dataset.
- o CGSHST A Monitor routine that creates the global memory section that contains tables shared by all the APCs and the monitor on a host.
- o MAPHST An MPU routine to map to the host's global section.

The NTM also uses the following system primitives to support the IISS applications:

- o ASCTIM Returns the host system time as an ASCII string.
- o CRTPRC Creates an instance of an IISS AP.
- o DELPRC Deletes a particular instance of an IISS AP.
- o GETNAM Returns the process name given to an AP by the NTM. This name is used to form an AP's mailbox name.
- o GETTIM Returns the system time as a binary string.
- o SGTPNM Returns the process name of a process's creating process.

While the table access routines do not directly use operating system calls, they may have some system dependency in the file access related code and should also be considered as a part of this section.

The Monitor portion of the NTM uses a set of system primitives to access the IISS console. These routines are INICON, RCVCON, GETCON, SNDCON, CNLCON, and TRMCON. They provide for reading and writing to the device used as the IISS console.

5.2 VAX Implementation of System Support Primitives

5.2.1 Process Control Modules

5.2.1.1 CRTPRC

CRTPRC is the module used by the NTM to create a new instance of an IISS AP. On the VAX, the VMS high level language operating system call "SYS\$CREPRC" is used. The call currently has the following structure:

```
CALL "SYS$CREPRC" USING BY REFERENCE PROCESS-ID
BY DESCRIPTOR IMAGE
BY DESCRIPTOR EQUIVNAME
BY DESCRIPTOR EQUIVNAME
BY DESCRIPTOR EQUIVNAME
BY VALUE 0
BY VALUE QUOTA-LIST
BY DESCRIPTOR PRONME
BY VALUE BASE-PRIORITY
BY VALUE 0
BY VALUE 0
BY VALUE STATUS-FLAG
GIVING SS-STATUS.
```

The descriptions of the arguments are in the VMS System services Manual [5]. The call is currently set up to run the created process as a subprocess of the NTM (the tenth argument set to zero gives a subprocess).

EQUIVNAME allows the subprocess to communicate with the terminal associated with the parent process. EQUIVNAME is obtained by a prior call to "SYS\$TRNLOG" on SYS\$COMMAND which gives the equivalence name for SYS\$COMMAND. On startup of the NTM, an assignment of SYS\$COMMAND to the current terminal completes this process [5]. This was useful for debugging in the development stages of the NTM.

5.2.1.2 DELPRC

The delete process routine, DELPRC, is called by the MPUs and is implemented by using the VAX operating system call "SYS\$FORCEX" as shown below.

```
CALL "SYS$FORCEX" USING BY VALUE 0
BY DESCRIPTOR PRONME
BY VALUE 3
GIVING SS-STATUS.
```

PRONME is the process name indicated by the MPU's call to DELPRC:

```
CALL DELPRC USING PRONME,  
NTM-RETURN,  
SS-STATUS.
```

The first argument indicates the process ID. By setting this to zero, VMS uses the process name (PRONME) to find the indicated process. The value of three allows an IISS AP to identify that it was forced to exit.

5.2.1.3 GETNAM

GETNAM finds the calling AP's process name by using the VMS "GET Job Process Information" call (SYS\$GETJPI).

A data structure ITEM-LIST is constructed as follows to indicate that only the process name information is required.

```
WORKING-STORAGE SECTION.  
01 ITEM-LIST.  
02 ITEM-PROCESS-NAME.  
03 PIC S9(4)    COMP VALUE 15.  
03 PIC S9(4)    COMP VALUE EXTERNAL JPI$-PRCNAM.  
03 POINTER      VALUE REFERENCE PROCESS-NAME.  
03 POINTER      VALUE REFERENCE PROCESS-NAME-LENGTH.  
02 TERMINATOR-ENTRY PIC S9(9) COMP VALUE ZERO.
```

The module's Procedure Division is shown below.

```
PROCEDURE DIVISION USING  APNAME,  
                           NTM-RETURN  
                           SS-STATUS.  
  
START-PROGRAM.  
    CALL "SYS$GETJPI" USING BY VALUE 1, 0, 0,  
                           BY REFERENCE ITEM-LIST,  
                           BY VALUE 0, 0, 0  
                           GIVING SS-STATUS.  
  
    IF SS-STATUS NOT EQUAL SS-NORMAL THEN  
        MOVE CALL-FAIL TO NTM-RETURN  
    ELSE  
        MOVE PROCESS-NAME TO APNAME  
        MOVE CALL-SUCCESSFUL TO NTM-RETURN.
```

5.2.1.4 SGTPNM

The SGTPNM routine is identical to the of GETNAM except it finds the parent process id of the requestor and uses that process id in its call to SYS\$GETJPI.

5.2.2 Global Memory Routine

The global memory implementation on the VAX uses one create section module, CGSHST, and a mapping module, MAPHST, that is called by the MPUs to map to the host global tables. This implementation is unwieldy because VMS COBOL requires manual page alignment of these memory sections. The caller (the monitor for CGSHST) of the create section module must have its global storage sections on page boundaries. Unfortunately, the memory allocation changes when new data or code is added to the MPU or monitor. Readjustments of these sections may be required when these modifications are made. The current technique for managing this problem is to use the debugger to find the real and the mapped addresses of the section. If they are different adjustments to the Buffer areas surrounding the sections in the MPUINI Module of the MPU are made until alignment is again attained. For example, the VAX monitor AP creates the host global section which has the following structure.

```
HOST GLOBAL SECTION      WORKING-STORAGE SECTION.
                          *
                          * INCLUDE FILES
                          *
                          COPY TBLHST OF IISSCLIB.
                          COPY TBLAPC OF IISSCLIB.
                          COPY TBLNST OF IISSCLIB.
                          COPY TBLLOG OF IISSCLIB.
                          COPY TBLCAT OF IISSCLIB.
                          COPY TBLDIR OF IISSCLIB.
                          COPY LOGSEL OF IISSCLIB.
                          01 DUMMY          PIC X(403).
                          01 GLOBAL-END     PIC X.
```

It calls CGSHST to create this section with the CALL:

```
CALL "CGSHST" USING IISS-ID,  
                    DIRECTORY-TABLE,  
                    HOST-STATUS-TABLE,  
                    AP-CLUSTER-STATUS-TABLE,  
                    LINK-STATUS-TABLE,  
                    LOGON-TABLE,  
                    MESSAGE-CATEGORY-TABLE,  
                    LOG-SELECT-STR,  
                    DUMMY,  
                    GLOBAL-END,  
                    RET-STATUS,  
                    SYSTEM-STATUS.
```

CGSHST uses the VMS System Call "SYS\$CRMPSC" shown below to create the section.

```
CALL "SYS$CRMPSC" USING BY REFERENCE IN-ADDRESS  
                        BY REFERENCE OUT-ADDRESS  
                        BY VALUE ACCESS-MODE  
                        BY VALUE 524297,  
                        BY DESCRIPTOR  
                        GLOBAL-SEC-NAME,  
                        BY REFERENCE VERS-IDENTIFICATION  
                        BY VALUE RELATIVE-PAGE  
                        BY VALUE CHANNEL  
                        BY VALUE PAGE-COUNT,  
                        BY VALUE VIRTUAL-BLK-NO  
                        BY VALUE 0, 0  
                        GIVING SS-STATUS.
```

IN-ADDRESS and OUT-ADDRESS are the real and mapped addresses and have the structures:

```
01 IN-ADDRESS.  
    05 IN-ADDR          PIC S9(9) COMP          OCCURS 2 TIMES.  
  
01 OUT-ADDRESS.  
    05 OUT-ADDR         PIC S9(9) COMP          OCCURS 2 TIMES.
```


To check for alignment, after the SYS\$CRMPSC call has been completed, IN-ADDR(1) should be compared to OUT-ADDR(1). These must be the same. Then IN-ADDR(2) should be compared to OUT-ADDR(2). These must be such that the mapped address (OUT-ADDR ADDR(2)) is in line with the DUMMY buffer of the section. This section created by the Monitor AP usually remains correctly aligned on code modifications.

Most of the problems seem to appear in the MPU mapping. Each MPU has its global sections defined in MPUINI.COB as:

```
*
WORKING-STORAGE SECTION.
*
*
* INCLUDE FILES
*
*!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
*** PLEASE LEAVE THESE WHERE THEY ARE -- AT TOP OF WORKING STG
*!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
  01 GLOBAL-PAGE-BOUNDRY PIC X(104).
*
  COPY TBLHST OF IISSCLIB.
  COPY TBLAPC OF IISSCLIB.
  COPY TBLLST OF IISSCLIB.
  COPY TBLLOG OF IISSCLIB.
  COPY TBLCAT OF IISSCLIB.
  COPY TBLDIR OF IISSCLIB.
  COPY LOGSEL OF IISSCLIB.
  COPY HSTGLE OF IISSCLIB.
***** END OF HOST GLOBAL MEMORY*****
*
*!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
```

The MPU's mapping call is the "map section" MAPHST call shown below.

```
CALL "MAPHST" USING HOST-STATUS-TABLE
                   AP-CLUSTER-STATUS-TABLE
                   LINK-STATUS-TABLE
                   LOGON-TABLE
                   MESSAGE-CATEGORY-TABLE
                   HOST-BUFFER
                   HOST-GLOBAL-END
                   RET-CODE
                   SS-STATUS.
```

The MAPHST routine uses the VMS service "SYS\$MGBLSC" to map to an existing section as:

```
CALL "SYS$MGBLSC" USING BY REFERENCE IN-ADDRESS,
                       BY REFERENCE OUT-ADDRESS,
                       BY VALUE ACCESS-MODE,
                       BY VALUE 000008,
                       BY DESCRIPTOR
                           GLOBAL-SEC-NAME,
                       BY REFERENCE VERS-IDENTIFICATION,
                       BY VALUE RELATIVE-PAGE,
                       GIVING SS-STATUS.
```

To complete the alignment check, the IN-ADDRESS and OUT-ADDRESS elements should be examined after the "SYS\$MGBLSC" call. If IN-ADDR(1) and OUT-ADDR(1) are not the same, APC-BUFFER should be adjusted so that HOST-STATUS-TABLE is on a page boundary. Again, OUT-ADDR(1) is a page boundary so adjustments must be made to this address or one 512 byte from it. The checks on IN-ADDR(2) against OUT-ADDR(2) should ensure that OUT-ADDR(2) is within the buffer zone at the end of each section.

5.2.2.1 CGSHST

The logic of CGSHST consists of a call to "GGSHST" by the monitor to create the host global section that is shared by the monitor and all the MPUs of that host. It contains IISS operating status information and is written to only by the monitor AP. The global section is created with the VAX "create global" system call, "SYS\$CRMPSC". The call requires the addresses of the beginning and end of the section. The addresses are obtained from an Assembler Program (ITMADR.MAR) that is called twice:

```
CALL "ITMADR" USING AP-CHAR-TABLE
                        GIVING GLOBAL-BEGIN-ADDR
CALL "ITMADR" USING GLOBAL-END
                        GIVING GLOBAL-END-ADDR
```

The assembler routine consists of the following statements:

```
ITEMADR::      .WORD 0
                MOVL 4(AP),RO
                RET
                .END
```

The arguments of the VAX "SYS\$CRMPSC" call are described in the VMS System Services Manual [5].

The call argument values are in the include file, HSTGLD.INC. The routine initializes the tables of the global section with statements that are in INTHST.INC.

5.2.2.2 MAPHST

MAPHST is the module called by the MPUs to map to the host global tables. The MPU CALL to MAPHST (Section 4.2.2) provides the structure that is being mapped to the host section. Like CGSHST, this routine uses the ITMADR entry (Section 4.2.2.1) to determine the beginning and ending addresses of his section. It then calls "SYS\$MGBLSC" (Section 4.2.2) whose argument values are in the WORKING STORAGE SECTION of the module. It returns the call's status to the calling MPU in SS-STATUS.

5.2.3 System Time Routines

5.2.3.1 ASCTIM

ASCTIM is the routine called by the MPUs for the ASCII system time. The module first uses the VMS call, "SYS\$GETTIM, to get the system time as a binary string and then calls "SYS\$ASCTIM" to convert this string to ASCII. The ASCII string is returned as a twenty three character string to the calling MPU in the format of DD-MMM-YYYY HH:MM:SS.TH.

5.2.3.2 GETTIM

GETTIM calls the VMS routine SYS\$GETTIM" as indicated below to get the system time in binary.

```
*
  PROCEDURE DIVISION USING      TIME-VALUE,
                                NTM-RETURN,
                                SS-STATUS.

*
  START-PROGRAM.
*
*
*   JUST GET THE TIME, SET NTM RETURN CODE AND RETURN
*
      CALL "SYS$GETTIM" USING  TIME-VALUE
                              GIVING SS-STATUS.
      IF SS-STATUS  = SS-NORMAL
          MOVE CALL-SUCCESSFUL TO NTM-RETURN
      ELSE
          MOVE CALL-FAILURE TO NTM-RETURN.
  EXIT-PROGRAM.
  EXIT PROGRAM.
```

APPENDIX A

TERMS AND ABBREVIATIONS

This paragraph contains definitions of IISS related terms and general computer terms that are used in this document.

AP	See Application Process.
Abnormal Termination	Stoppage of a process prior to its normal completion. (Abort)
Termination	(Abort)
Alive AP	An application process that is capable of being initiated.
Application Process	Within the IISS Architecture; a cohesive unit of software that can be initiated as a unit to perform some function or functions. See also: Process.
Application Process Cluster	A logically related group of application processes that reside on a single host system. The application processes are collected at a single cluster because of their common need to access the same database. (In previous NTM documents this concept was referred to as "Workstation").
Authorization	A relationship between message type and legal paths of messages conforming to that type.
Awake AP	An application process that is currently executing.
Backlog	Messages waiting, in a queue, to be processed.
CDMRP	See Common Data Model Request Processor.
COMM	See Communications Handler.
Clean Point	A baseline of the entire system state at a given point in time so that recovery is possible from that point.
Cluster	See Application Process Cluster.
Common Data	Information that may be shared among users and is subject to policies and procedures of the IISS.

Common Data Model	A description of the data, its structure, allowable (CDM) operations, and integrity constraints for data of common interest within the IISS.
Common Data Model Request Processor	The implementation of the CDM within the IISS. The current design specifies the use of extremities.
Communications Handler	An IISS Configuration Item that services IISS network communications. It accepts messages from the NTM for off-host destinations. It provides NTM with messages from off-host sources.
DBMS	See Data Base Management System.
DML	See Data Manipulation Language.
Database	A collection of logically related data managed by DBMS.
Data Base Management System	A computerized system consisting of numerous components, which have as their collective purpose the implementation, management, and protection of large-scale data bases.
Data Manipulation Language	A formal language for specifying data storage, modification, and retrieval functions under a DBMS.
Destination	An application process (which may be a user role) to which a message is sent.
Heterogeneous	Composed of parts of different kinds; having widely dissimilar elements or constituents.
Homogeneous	Composed of parts all of the same kind.
Host	See Host Machine.
Host Machine	A configuration of a processor(s) and associated peripherals.
IISS Component	Application Processes involved in IISS support activities. See: CDMRP, UI, COMM.
IISS Operation Messages	Messages entered directly to the NTM by an operator. These include: start-up, shut-down, restart, and provide status and statistics.
IISS Shut-down	A signal or IISS operator command that initiates the orderly shut-down of the entire IISS system or an individual workstation.

IISS Start-up	A signal or IISS operator command that initiates the IISS on a single host.
Integrated AP	An application process designed and written in accordance with the IISS Integration rules.
Initiation Message	A message that specifically requires the initiation of an application process. The message may also carry data for that process.
LAN	Local Area Network.
Load	To bring a bound (linked) set of computer instructions into a computer memory unit in preparation for its execution as a process.
MDL	Message Definition Language. Neutral syntax and semantics for formatting IISS messages.
MM	See Message Manager.
Maintain Operability	The name given to that part of the NTM that will facilitate system wide services of restart, recovery, shut-down, and system status monitoring and recording.
Message	A structured unit of information.
Message and Error Log	The record of messages (legal and erroneous) that are processed by the message manager.
Message Category	A group of message types sharing common processing requirements.
Messages from Off-Cluster	Any message that has as its source an AP NTM that is not resident on the cluster in question.
Message from On-Cluster	Any message sent by a local AP that is directed to another AP or the APC or to another NTM (i.e., requiring the routing services of the the Message Manager).
Message, Definition	The legal (recognized and allowed) values to which a message must conform to be accepted for processing at an AP on a given APC on a given host.
Message Manager	The name given to the part of the NTM that provide the service for the identified Manage Message functions.

Message to IISS Operator	Typically a response to an operation request. This message could also be a request for operator action.
Messages to Off-Cluster	Any message that has as its source an AP or NTM that is not resident on the cluster in question.
Message to On-Cluster	Messages directed to an AP (resident on the cluster) that contain information necessary for the process to perform.
Message to be Routed	A message that has another AP as its destination.
Message Type	The identification of the nature of a given message.
MO	See Maintain Operability.
MO Data and Status Messages	Any message directed to the maintain operability function.
Native Mode	Character code native to a particular machine, either ASCII or EBCDIC.
Network	Several computers and a communications facility connecting them.
Non-Integrated AP	An Application Process designed without adherence to the IISS Integration rules.
NTM	Network Transaction Manager. That portion of the IISS that manages messages and application processes and maintains the operability of the IISS.
OS	See Operating System.
OS Process Control Response	Response from the host operating system to a request for services, (i.e., process ID assigned, process exists, process does not exist).
OS Process Control Request	Request for services to be provided by the Request host operating system.
On-APC AP Messages	Any message that has as its destination an AP that is resident on the workstation in question.
Operability Messages	Messages initiated by Maintain Operability to handle an IISS operation request and to monitor the IISS system.

Operating System	Software that controls the execution of computer programs and that may provide scheduling, debugging, input-output control, accounting, compilation, storage assignment, data management, and related overall system management.
Priority	The expectation of processing urgency assigned to a message type. The recognized transaction priorities are: standard, immediate and time-triggered.
Process	The basic unit of work from the stand-point of the computer's operating system.
Process Manager	The name given to the part of the NTM that will provide the service for the identified Manage Processes function. (See DS/A2, Section 10.2)
Process Name	The AP Name and instance identifier of an initiated AP.
Requirement	Stated functions for information processing and associated constraints.
Resources	People, hardware, software, and other component used to process information.
Source	An Application Process from which a message is sent.
Status and Statistics Log	Data regarding the AP status, resource utilization, message traffic provided in response to an operation request.
System	A collection of people, hardware, software and methods organized to accomplish a set of specific functions.
Task	See Process.
Test Bed	A collection of computer hardware, software, storage devices, and other peripherals used for testing application software and software and system concepts. Usually not the target operational system.
UI	See User Interface.
Update	The process of changing values in all or selected entries, groups, or data items stored in a database or adding or deleting data occurrences.

User	Human being who requests processing to be done. The person entering a transaction at a terminal. The user can also be a non-human that performs these functions but this term will not be used in this sense in this specification.
User Application Process	Any application process not involved in IISS support activities.
User Interface	An application process that manages the user terminal interface.
View	That which is within the range of vision. Also, that data which is of interest to a specific application process.
Workstation	See Application Process Cluster

APPENDIX B

NTM DOCUMENTATION

This document is intended to provide information on the internals of the NTM. However, the information presented represents only the tip of the proverbial iceberg. Further details are to be found in the following:

- [1] General Electric, "IISS Volume VI, Network Transaction Manager Subsystem, Part 1, Development Specification," 31 May 1988. (DS620342000).

This document provides the functional description of the NTM components. It includes a data dictionary of data items used by the NTM along with a full description of each of the internal message types.

- [2] General Electric, "IISS Volume VI, Network Transaction Manager Subsystem, Parts 5, 6, and 7, Product Specification." 31 May 1988. (PS620342100, PS620342200, and PS620342300).

Published in three volumes, this document contains the structure charts and module descriptions of the NTM code "As-Built." Volume\1 covers the Monitor AP; Volume 2 covers the Message Processing Unit; and Volume 3 contains the NTM services, System Dependent Code, and Table Handling Routines.

- [3] B.M.A.C., "ICAM, IDBM, Test Bed, Network Transaction Manager Operator's Manual (Release 2.2)," 31 May 1988. (OM 620342000)

This document is intended for use by the IISS Operator. It includes descriptions of the Operator Commands and IISS error codes along with guidelines for table maintenance and trouble shooting suggestions.

- [4] B.M.A.C., "ICAM, IDBM, Test Bed, Network Transaction Manager Programmer's Manual," Release 2.2. 31 May 1988. (PRM620342000)

This document is intended for use by Application developers writing AP's that will run in the IISS environment. It contains full descriptions of the system services including calling parameters, inputs, outputs, and return code values. Examples are provided along with suggestions for use.

- [5] Digital Equipment Corporation; "VAX/VMS System Services Reference Manual"; July 1985.